On the correct use of the negation map in the Pollard rho method

Entirely boring, enforced talk

Daniel J. Bernstein, Tanja Lange, Peter Schwabe

Eindhoven University of Technology

October 18, 2010

ECC 2010 Rump Session

Boring?

Yep, everybody here knows what I'm going to tell you.

Enforced?

1. Rump-session chairs are coauthors,
2. rump-session chairs are my supervisors, and
3. I have to submit my thesis end of this week.

They want this talk.

Speeding up Pollard rho by a factor of $\sqrt{2}$

- Pollard's rho algorithm is the best known algorithm to solve "hard" ECDLPs
- Use pseudo-random walk in G through $P_{i+1} = f(P_i)$
- Solve ECDLP when walk collides
- Expected number of iterations: $\sqrt{\pi |G|/2}$

Speeding up Pollard rho by a factor of $\sqrt{2}$

- Pollard's rho algorithm is the best known algorithm to solve "hard" ECDLPs
- Use pseudo-random walk in G through $P_{i+1} = f(P_i)$
- Solve ECDLP when walk collides
- Expected number of iterations: $\sqrt{\pi |G|/2}$
- Idea: Define walk on equivalence classes of efficiently computable endomorphisms
- For elliptic curves: negation
- Simply choose "smallest" representative modulo negation
- Save factor of $\sqrt{2}$ in the number of iterations
- This is a textbook optimization

Current record for solving ECDLPs

- In July 2009 Bos, Kaihara, Kleinjung, Lenstra, and Montgomery announced that they solved a 112-bit ECDLP using a cluster of 200 PlayStation 3 gaming consoles
- Interesting fact: They did not use the $\sqrt{2}$ speedup from the negation map

Current record for solving ECDLPs

- In July 2009 Bos, Kaihara, Kleinjung, Lenstra, and Montgomery announced that they solved a 112-bit ECDLP using a cluster of 200 PlayStation 3 gaming consoles
- Interesting fact: They did not use the $\sqrt{2}$ speedup from the negation map
- Reason: When you have 200 PlayStations sitting around, you don't care.

Current record for solving ECDLPs

- In July 2009 Bos, Kaihara, Kleinjung, Lenstra, and Montgomery announced that they solved a 112-bit ECDLP using a cluster of 200 PlayStation 3 gaming consoles
- Interesting fact: They did not use the $\sqrt{2}$ speedup from the negation map
- Reason: "We did not use the common negation map since it requires branching and results in code that runs slower in a SIMD environment."

Slower?

- Common way to construct iteration function f:
  - Precompute points $T_0, \ldots, T_k$,
  - define function $h : G \to \{0, \ldots, k\}$
  - define $f(P)$ as $P + T_{h(P)}$

Slower?

- Common way to construct iteration function f:
  - Precompute points $T_0, \ldots, T_k$,
  - define function $h : G \to \{0, \ldots, k\}$
  - define $f(P)$ as $P + T_{h(P)}$
- Problem with negation: fruitless cycles, $P \to -(P + T_i) \to P$
- Several techniques to resolve these fruitless cycles, but annoying with SIMD

annoying $\neq$ impossible

TU/e Technische Universiteit Eindhoven University of Technology

- New implementation by the rump-session chairs and me
- Solves the same ECDLP 1.8 times faster (expected)
- Speedup of almost $\sqrt{2}$ by using the negation map
- Branchless computation
- Remaining factor of ≈ 1.3 from faster arithmetic
- Paper will be online very soon (this week)

Conclusion

Use of the negation map in Pollard's rho algorithm to solve the ECDLP gives a speedup of a factor of $\approx \sqrt{2}$.