Concrete quantum cryptanalysis of binary elliptic curves

Gustavo Banegas, Daniel J. Bernstein, Iggy van Hoof and Tanja Lange

2021, September 13
Google Claims ‘Quantum Supremacy,’ Marking a Major Milestone in Computing

Quantum computing leaps ahead in 2019 with new power and speed

- Shor (1994): Sufficiently large quantum computers break DLP and RSA
- How big do these quantum computers need to be?
- Previous recent work: RSA & prime field ECDH.
- This work: binary ECDH.
- Results: $7n + \lfloor \log_2(n) \rfloor + 9$ qubits, $48n^3 + O(n^{\log_2(3)+1})$ TOF gates.
Most expensive step in Shor is adding precomputed points $[2^i]P$.
We treat rest of Shor as blackbox.
Point addition uses operations in $\mathbb{F}_{2^n}$:
- Addition
- Multiplication
- Division
How do we build quantum circuits?
Quantum Gates

- Quantum bits: qubits.
- Today quantum computing 101: no purely quantum gates.
- Classical reversible gates:

\[
\begin{align*}
\text{NOT:} & \\
& a \quad 1 - a \\
\text{CNOT:} & \\
& a \quad a \oplus b \\
& b \quad b
\end{align*}
\]

\[
\begin{align*}
\text{SWAP:} & \\
& a \quad b \\
& b \quad a
\end{align*}
\]

\[
\begin{align*}
\text{TOF:} & \\
& a \\
& b \\
& c \quad c \oplus (a \cdot b)
\end{align*}
\]
Quantum circuits

- We can now make circuits.
- Number of qubits is most important.
- Need a measure of quality:
  - Gate count?
  - TOF gates?
  - Depth?
  - Physical implementation?
Binary addition with a constant: NOT gates (same as classically).

Binary addition of 2 variables:
- bitwise XOR $\rightarrow$ CNOT.
- Reversible: 2 inputs $f, g$; 2 outputs $f \oplus g, g$.
- $n$ CNOTs.
Basic arithmetic: Multiplication by $x$ in $\mathbb{F}_{2^n}$

- Field: use $\mathbb{F}_{2^n} \cong \mathbb{F}_2[x]/m(x)$ for an irreducible polynomial $m(x) \in \mathbb{F}_2[x]$ of degree $n$.
- Times $x$ without reduction is free with SWAP.
- Modular reduction in 1 CNOT for trinomial $m(x)$.
- Modular reduction in 3 CNOTs for pentanomial $m(x)$.
- Do in reverse for division by $x$.

$$|g_0\rangle \quad \cdots \quad |g_3\rangle$$

Figure: Multiplication by $x$ modulo $x^4 + x + 1$ with $g_0 + \cdots + g_3x^3$ as the input and $h_0 + \cdots + h_3x^3 = x \cdot g \mod x^4 + x + 1 = g_3 + (g_0 + g_3)x + g_1x^2 + g_2x^3$ as the output.
Basic arithmetic: Multiplication by constant & Squaring in $\mathbb{F}_{2^n}$

- Multiplication by a constant is a linear map.
- Turn linear map into a series of CNOTs using LUP decomposition.
- Do the same with squaring, linear map; $(a + b)^2 = a^2 + b^2$ in $\mathbb{F}_{2^n}$.
- Alternatively, adding the squaring result to a second polynomial also with only CNOTs.
Advanced arithmetic: Multiplication in $\mathbb{F}_{2^n}$

- Earlier work (van Hoof, Quantum Information and Computation 2020):
  - Quantum Karatsuba multiplication in $\mathbb{F}_{2^n}$.
  - No \textit{ancillary} qubits needed, only $3n$ space.
  - Previous work used extra qubits.
  - Optimal TOF gate count for Karatsuba: $n^{\log_2 3}$ TOF gates.
Most expensive step: division or inversion.
We compare 2 methods:
- Extended Euclidean algorithm.
- Fermat’s little theorem.
Division: Extended Euclidean algorithm

- Normal Euclidean algorithm has variable number of steps.
- Based on constant time classical inversion (Bernstein & Yang, CHES 2019).

\[
|\delta\rangle \quad \frac{1}{|\delta|} \quad +1 \quad |\delta\rangle \\
|\text{sign}\rangle \quad |\text{sign}\rangle \\
|f\rangle \quad \frac{1}{n+1} \quad \times \quad |f\rangle \\
|g\rangle \quad \frac{1}{n+1} \quad \times \quad |g\rangle \\
g_0[\ell] = |0\rangle \\
a = |0\rangle \\
|r\rangle \quad \frac{1}{n+1} \\
|v\rangle \quad \frac{1}{n+1} \quad \cdot \chi \quad |v\rangle \\
\]

Figure: Step \(\ell\) of Algorithm 1. \(|\delta| = \lfloor \log(n) \rfloor + 1\).
Fermat’s little theorem: \( x^p \equiv x \mod p \rightarrow x^{p-2} \equiv x^{-1} \mod p \).

Binary FLT: \( x^{2^n-2} \equiv x^{-1} \mod m(x) \).

Itoh-Tsujii inversion optimizes this.

Large number of squarings, low number of multiplications.

Number of multiplications depends on HW of \( n - 1 \).

Every multiplication costs \( n \log_2 3 \) TOF gates.

Squaring costs only CNOT gates.
**FLT-based inversion circuit**

Figure: Step 1-3 of Algorithm 2 for $n = 10$. $S$ is the squaring circuit and $M$ is multiplication.
**XGCD vs FLT**

- Extended Euclidean algorithm uses more TOF gates.
- Fermat’s little theorem uses more qubits and CNOT gates.
- Example: $n = 233$:

| inversion method | TOF gates | qubits |
|------------------|-----------|--------|
| XGCD             | 827,977   | 1,646  |
| FLT              | 132,783   | 3,029  |
Point addition

- On the curve we need to add multiples of $P$ to quantum $P_1 = (x_1, y_1)$.
  - If qubit $q_i = 1$ output $(x_3, y_3) = P_1 + P_2$ with pre-computed $P_2 = (x_2, y_2) = [2^i]P$.
  - Else: output $P_1$.

- Binary addition in affine coordinates: 2 S, 2 M and 2 D in $\mathbb{F}_{2^n}$.
- 2\textsuperscript{nd} division returns ancillary qubits to all-0.

- Special cases:
  - Addition with $O$.
  - Addition when $x_1 = x_2$. 

\[
\begin{align*}
|x_1\rangle & \quad /n \cdot +x_2 \quad +a + x_2 \quad +x_2 \quad \text{or} \quad |x_3\rangle \quad \text{or} \quad |x_1\rangle \\
|q\rangle & \quad \text{or} \quad |q\rangle \\
|y_1\rangle & \quad /n \cdot +y_2 \quad M \quad S \quad S \quad M \quad +y_2 \quad \text{or} \quad |y_3\rangle \quad \text{or} \quad |y_1\rangle \\
|0\rangle & \quad /n \cdot D \quad \text{or} \quad |0\rangle
\end{align*}
\]
Can we use classical precomputation?

Classical computation is very cheap.

Classically precompute all \(2^j \ell (a_0 P + a_1 2P + \cdots + a_{\ell-1} 2^{\ell-1} P)\) with \(a_i \in \{0, 1\}\) and handle \(\ell\) bits in one quantum addition.

Need qRAM lookups.

Example: \(n = 233\):

| Window size | Point additions | TOF gates | Lookups | Pre-computed points |
|-------------|----------------|-----------|---------|---------------------|
| 1           | 468            | 781 M     | 0       | 0                   |
| 7           | 68             | 113 M     | 408     | 8,704               |
| 16          | 30             | 52 M      | 180     | 1,966,080           |
| 32          | 16             | 27 M      | 96      | 68,719,476,736      |
### Summary: No windowing

- Division is the most expensive step.
- Results without windowing:

| n  | qubits | Point addition | Total          |
|----|--------|----------------|----------------|
|    |        | TOF gates      | TOF gates      |
| 8  | 68     | 7,360          | 132,480        |
| 16 | 125    | 21,016         | 714,544        |
| 127| 904    | 559,141        | 143,140,096    |
| 163| 1,157  | 893,585        | 293,095,880    |
| 233| 1,647  | 1,669,299      | 781,231,932    |
| 283| 1,998  | 2,427,369      | 1,378,745,592  |
| 571| 4,015  | 8,987,401      | 10,281,586,744 |
Summary: Windowing

- Need approximation of cost of qRAM lookup.
- Previous work: $2(2^\ell - 1)$ TOF gates per lookup (Babbush, Gidney, Berry, Wiebe, McClean, Paler, Folwer and Neven, Physical Review, 2018).
- Results with windowing:

| $n$ | $\ell$ | TOF gates | Lookups | Total TOF gates | pre-computed points |
|-----|--------|-----------|---------|-----------------|---------------------|
| 8   | 7      | 29,344    | 24      | 35,440          | 512                 |
| 16  | 8      | 125,808   | 36      | 144,168         | 1,536               |
| 127 | 13     | 11,733,960| 120     | 13,699,800      | 163,840             |
| 163 | 13     | 24,113,592| 156     | 26,669,184      | 212,992             |
| 233 | 14     | 58,401,000| 204     | 65,085,264      | 557,056             |
| 283 | 14     | 101,913,840| 252   | 110,170,872     | 688,128             |
| 571 | 16     | 655,955,224| 432   | 712,577,464     | 4,718,592           |
Comparison to other work

- Division and multiplication numbers look good for binary fields.
- General results:
  - \(7n + \lfloor \log_2(n) \rfloor + 9\) qubits, mostly ancillary qubits for division.
  - \(48n^3 + 8n^{\log_2(3)+1} + 352n^2 \log_2(n) + 512n^2 + O(n^{\log_2(3)})\) TOF gates.
- Prime field: similar results (Roetteler, Naehrig, Svore and Lauter, Asiacrpt 2017 & Häner, Jaques, Naehrig, Roetteler and Soeken, PQCrypto 2020).
  - Lots of speedup over the prime field case: addition, multiplication, division all cheaper in \(\mathbb{F}_{2^n}\).
- Projective binary coordinates: not optimized for space (Amento, Roetteler, Steinwandt, Quantum Information and Computation 2013)
  - Significantly worse space.
  - Lower TOF gate count due to fewer divisions.