Quantum Breakthrough: Factoring with Shors Algorithm

How Quantum Computing Breaks Classical Cryptography

Taslim Haroun

April 2025

The Problem: Integer Factorization

- **RSA encryption** relies on the difficulty of factoring large numbers
- Example: $15 = 3 \times 5$, but $221 = 13 \times 17$ takes effort
- Best classical algorithm: Number Field Sieve sub-exponential time
- For 2048-bit numbers: *millions of years* on classical computers
- **Challenge**: Can quantum computing do better?

Enter Peter Shor (1994)

- **Peter Shor**, Bell Labs, 1994
- **Idea**: Use quantum superposition and interference to find periods of functions
- **Reduction**: Factoring $\rightarrow$ Order-finding $\rightarrow$ Period of $f(x) = a^x \mod N$
- **Result**: Polynomial-time quantum algorithm
- **Impact**: Threatens RSA, ECC, and modern public-key cryptography

Classical vs Quantum Approach

	Classical	**Quantum**
Factor 15	Fast	Same
Factor 2048-bit N	Infeasible ($\sim 10^{300}$ years)	Hours/Days (theoretically
Core Idea	Try divisors	Find period using QFT
Complexity	Sub-exponential	$O((\log N)^3)$

Quantum advantage grows with number size

The Math Behind Shor

Given N, pick random $a < N$ with $\gcd(a, N) = 1$
Define: $f(x) = a^x \mod N \to$ periodic with period r (order of $a \mod N$)
If r is even and $a^{r/2} \not\equiv -1 \mod N$, then:

$$p = \gcd(a^{r/2} + 1, N), \quad q = \gcd(a^{r/2} - 1, N)$$

are nontrivial factors.

Example: $N = 15$, $a = 7$

$$\begin{aligned} 7^1 &\mod 15 = 7 \\ 7^2 = 49 &\mod 15 = 4 \\ 7^3 = 28 &\mod 15 = 13 \\ 7^4 = 91 &\mod 15 = 1 \to r = 4 \end{aligned}$$

Then: $7^2 = 4$, so $\gcd(4 + 1, 15) = 5$, $\gcd(4 - 1, 15) = 3$ ✓

Quantum Circuit Overview

Registers:

- Input: t qubits $\rightarrow$ superposition of x
- Output: $\log_2(N)$ qubits $\rightarrow$ store $f(x) = a^x \mod N$

Steps:

1. Prepare $|0\rangle|0\rangle$
2. Apply H gates $\rightarrow \sum |x\rangle|0\rangle$
3. Apply U^{2^j} controlled gates $\rightarrow \sum |x\rangle|f(x)\rangle$
4. Measure output $\rightarrow$ collapse to periodic x
5. Apply Inverse QFT on input $\rightarrow$ peaks at multiples of $2^t/r$
6. Measure $\rightarrow$ estimate r

Uses quantum parallelism and interference

Quantum Fourier Transform (QFT)

- **QFT**: Quantum analog of FFT
- Transforms time $\rightarrow$ frequency domain
- Turns periodic states into sharp peaks
- Enables efficient period extraction

QFT Circuit:

- Hadamards + controlled phase rotations
- Depth: $O(n^2)$

Simulation Results (N=15)

Implementation:

- Qiskit on quantum simulator
- $a = 7$, $N = 15$, $t = 3$ qubits
- Measured: $y = 0, 2, 4, 6 \rightarrow$ multiples of 2
- Estimate: $r = 4$

Output:

- Measurement counts: {'000': 240, '010': 260, '100': 255, '110': 270}
- Estimated period $r = 4$
- Factors: $\gcd(7^2 \pm 1, 15) = 3$ and 5 ✓

Even on simulator, correct period found with high probability

Challenges & Real-World Status

Current Limitations:

- Requires fault-tolerant quantum computers
- Needs thousands of logical qubits
- NISQ devices cannot run full Shor yet

Progress:

- 2001: IBM factored 15 on 7-qubit NMR quantum computer
- 2012: 21 factored using photonic chip
- Today: Simulations dominate; real factorization still limited

Crypto Impact: Future quantum computers will break RSA unless we migrate to post-quantum cryptography (PQC)

Conclusion & Future

Key Takeaways:

- Shors algorithm is a killer app for quantum computing
- Proves quantum advantage in computational complexity
- Demonstrates power of superposition, entanglement, and interference

Future:

- Hybrid algorithms
- Error-corrected quantum processors
- Migration to lattice-based crypto

Call to Action:

- Start learning quantum programming today the future is superpositional!

Qiskit Code: github.com/tahslim/shor-algorithm-demo