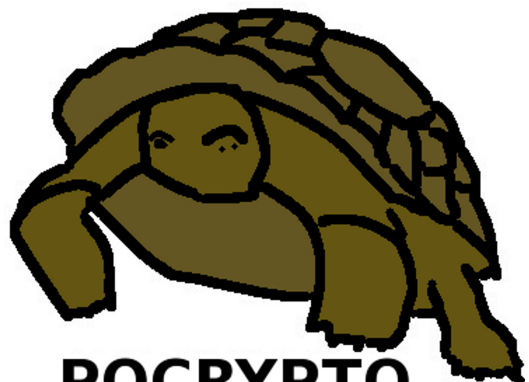
Practical Post-Quantum Cryptography

Tanja Lange



31 Jul 2017

SIAM-AG 2017

Algorithms for Quantum Computation: Discrete Logarithms and Factoring

Peter W. Shor
AT&T Bell Labs
Room 2D-149
600 Mountain Ave.
Murray Hill, NJ 07974, USA

Abstract

A computer is generally considered to be a universal computational device; i.e., it is believed able to simulate any physical computational device with a cost in computation time of at most a polynomial factor. It is not clear whether this is still true when quantum mechanics is taken into consideration. Several researchers, starting with David Deutsch, have developed models for quantum mechanical computers and have investigated their computational properties. This paper gives Las Vegas algorithms for finding discrete logarithms and factoring integers on a quantum computer that take a number of steps which is polynomial in the input size, e.g., the number of digits of the integer to be factored. These two problems are generally considered hard on a classical computer and have been used as the basis of several proposed cryptosystems. (We

[1, 2]. Although he did not ask whether quantum mechanics conferred extra power to computation, he did show that a Turing machine could be simulated by the reversible unitary evolution of a quantum process, which is a necessary prerequisite for quantum computation. Deutsch [9, 10] was the first to give an explicit model of quantum computation. He defined both quantum Turing machines and quantum circuits and investigated some of their properties.

The next part of this paper discusses how quantum computation relates to classical complexity classes. We will thus first give a brief intuitive discussion of complexity classes for those readers who do not have this background. There are generally two resources which limit the ability of computers to solve large problems: time and space (i.e., memory). The field of analysis of algorithms considers the asymptotic demands that algorithms make for these resources as a function of the problem size. Theoretical

Universal quantum computers are coming, and are scary

- Shor's algorithm solves in polynomial time:
  - Integer factorization. RSA is dead.
  - The discrete-logarithm problem in finite fields. DSA is dead.
  - The discrete-logarithm problem on elliptic curves. ECDSA is dead.
- This breaks all current public-key cryptography on the Internet!
- Massive research effort. Tons of progress summarized in, e.g., https://en.wikipedia.org/wiki/Timeline_of_quantum_computing.

Universal quantum computers are coming, and are scary

- Shor's algorithm solves in polynomial time:
  - Integer factorization. RSA is dead.
  - The discrete-logarithm problem in finite fields. DSA is dead.
  - The discrete-logarithm problem on elliptic curves. ECDSA is dead.
- This breaks all current public-key cryptography on the Internet!
- Massive research effort. Tons of progress summarized in, e.g., https://en.wikipedia.org/wiki/Timeline_of_quantum_computing.
- Mark Ketchen, IBM Research, 2012, on quantum computing: "We re actually doing things that are making us think like, 'hey this isn't 50 years off, this is maybe just 10 years off, or 15 years off.' It's within reach."

Universal quantum computers are coming, and are scary

- Shor's algorithm solves in polynomial time:
  - Integer factorization. RSA is dead.
  - The discrete-logarithm problem in finite fields. DSA is dead.
  - The discrete-logarithm problem on elliptic curves. ECDSA is dead.
- This breaks all current public-key cryptography on the Internet!
- Massive research effort. Tons of progress summarized in, e.g., https://en.wikipedia.org/wiki/Timeline_of_quantum_computing.
- Mark Ketchen, IBM Research, 2012, on quantum computing: "We re actually doing things that are making us think like, 'hey this isn't 50 years off, this is maybe just 10 years off, or 15 years off.' It's within reach."
- Also, Grover's algorithm speeds up brute-force searches.
- Example: Only 2^{64} quantum operations to break AES-128; 2^{128} quantum operations to break AES-256.

Even higher urgency for long-term confidentiality

- Attacker can break currentlyused encryption (ECC, RSA) with a quantum cmputer.
- Even worse, today's encrypted communication is being stored by attackers and will be decrypted years later with quantum computers. All data can be recovered in clear from recording traffic and breaking the public key scheme.
- How many years are you required to keep your data secret? From whom?

- Signature schemes can be replaced once a quantum computer is built – but there will not be a public announcement

Even higher urgency for long-term confidentiality

- Attacker can break currentlyused encryption (ECC, RSA) with a quantum cmputer.
- Even worse, today's encrypted communication is being stored by attackers and will be decrypted years later with quantum computers. All data can be recovered in clear from recording traffic and breaking the public key scheme.
- How many years are you required to keep your data secret? From whom?

- Signature schemes can be replaced once a quantum computer is built – but there will not be a public announcement . . . and an important function of signatures is to protect operating system upgrades.
- Protect upgrades *now* with post-quantum signatures.

History of post-quantum cryptography

- 2003 Daniel J. Bernstein introduces term Post-quantum cryptography.
- PQCrypto 2006: International Workshop on Post-Quantum Cryptography.

History of post-quantum cryptography

- 2003 Daniel J. Bernstein introduces term Post-quantum cryptography.
- PQCrypto 2006: International Workshop on Post-Quantum Cryptography.
- PQCrypto 2008, PQCrypto 2010, PQCrypto 2011, PQCrypto 2013.
- 2014 EU publishes H2020 call including post-quantum crypto as topic.
- ETSI working group on "Quantum-safe" crypto.
- PQCrypto 2014.
- April 2015 NIST hosts first workshop on post-quantum cryptography
- August 2015 NSA wakes up

NSA announcements

August 11, 2015

IAD recognizes that there will be a move, in the not distant future, to a quantum resistant algorithm suite.

NSA announcements

August 11, 2015

> *IAD recognizes that there will be a move, in the not distant future, to a quantum resistant algorithm suite.*

August 19, 2015

> *IAD will initiate a transition to quantum resistant algorithms in the not too distant future.*

NSA comes late to the party and botches its grand entrance.

NSA announcements

August 11, 2015

> *IAD recognizes that there will be a move, in the not distant future, to a quantum resistant algorithm suite.*

August 19, 2015

> *IAD will initiate a transition to quantum resistant algorithms in the not too distant future.*

NSA comes late to the party and botches its grand entrance.

Worse, now we get people saying "Don't use post-quantum crypto, the NSA wants you to use it!".

NSA announcements

August 11, 2015

> *IAD recognizes that there will be a move, in the not distant future, to a quantum resistant algorithm suite.*

August 19, 2015

> *IAD will initiate a transition to quantum resistant algorithms in the not too distant future.*

NSA comes late to the party and botches its grand entrance.

Worse, now we get people saying "Don't use post-quantum crypto, the NSA wants you to use it!". Or "NSA says NIST P-384 is post-quantum secure".

NSA announcements

August 11, 2015

> *IAD recognizes that there will be a move, in the not distant future, to a quantum resistant algorithm suite.*

August 19, 2015

> *IAD will initiate a transition to quantum resistant algorithms in the not too distant future.*

NSA comes late to the party and botches its grand entrance.

Worse, now we get people saying "Don't use post-quantum crypto, the NSA wants you to use it!". Or "NSA says NIST P-384 is post-quantum secure". Or "NSA has abandoned ECC."

Post-quantum becoming mainstream

- PQCrypto 2016: 22–26 Feb in Fukuoka, Japan, $>$ 200 people

- NIST is calling for post-quantum proposals; submissions due Nov 2017.
- https://2017.pqcrypto.org/ events in NL
  - Jun 19 – 23 PQCRYPTO school (Eindhoven)
  - Jun 22 – 23 ECRYPT-CSA Executive school (Eindhoven)
  - Jun 26 – 28 PQCrypto (Utrecht)

Flush, Gauss, and Reload A Cache-Attack on the BLISS Lattice-Based Signature Scheme

Leon Groot Bruinderink, Andreas Hülsing,
Tanja Lange, and Yuval Yarom

Analyzing deployed systems

- Bimodal Lattice Signature Scheme (BLISS) (CRYPTO '13 by Léo Ducas and Alain Durmus and Tancrède Lepoint and Vadim Lyubashevsky)
- Pretty short and efficient; already included in strongSwan (library for IPsec-based VPN).
- Needs noise from discrete Gaussian distribution. (Discrete Gaussian: restrict continuous Gaussian to integer values, scale probabilties to sum up to one.)
- Obvious issues about constant time – but hard to exploit cumulative measurement. What can we actually attack?
- Our paper (CHES 2016) was first side-channel attack on a lattice-based signature scheme.
- Follow-up work (Leon, Yuval, Peter Pessl) even breaks a deployed version.

Background

- Work in $R = \mathbf{Z}[x]/(x^n + 1)$, $n = 2^r$ (typically $n = 512$), and $R_q = (\mathbf{Z}/q)[x]/(x^n + 1)$ for q prime, e.g., $q = 12289$.
- Switch representation between polynomial and vector notation.

$$f(x) = \sum_{i=0}^{n-1} f_i x^i \Leftrightarrow f = (f_{n-1}, f_{n-2}, \ldots, f_1, f_0).$$

- Polynomial multiplication then corresponds to vector-matrix multiplication. Let $f, g, \in R_q$, then

$$f \cdot g = fG = gF,$$

where $F, G \in (\mathbf{Z}/q)^{n \times n}$ match vectors of $x^i f$ and $x^j g$:

$$F = \begin{pmatrix} f_0 & -f_{n-1} & -f_{n-2} & \cdots & -f_1 \\ f_1 & f_0 & -f_{n-1} & \cdots & -f_2 \\ \vdots & \vdots & \ddots & \vdots & \\ f_{n-1} & f_{n-2} & f_{n-3} & \cdots & f_0 \end{pmatrix}$$

Simplified BLISS

- Secret key $S = (s_1, s_2) = (f, 2g + 1) \in R_q^2$, f, g sparse in $\{0, \pm 1\}^n$.
- Public key $A = (a_1, a_2) \in R_{2q}^2$, with key equation $a_1 s_1 + a_2 s_2 \equiv q \bmod 2q$.
- Computed as $a_q = (2g + 1)/f \bmod q$ (restart if f is not invertible); then $A = (2a_q, q - 2) \bmod 2q$.

Simplified BLISS

- Secret key $S = (s_1, s_2) = (f, 2g + 1) \in R_q^2$, f, g sparse in $\{0, \pm 1\}^n$.
- Public key $A = (a_1, a_2) \in R_{2q}^2$, with key equation $a_1 s_1 + a_2 s_2 \equiv q \bmod 2q$.
- Computed as $a_q = (2g + 1)/f \bmod q$ (restart if f is not invertible); then $A = (2a_q, q - 2) \bmod 2q$.
- Can verify key guess for f with key equation; g computable.
- Focus on s_1; $-S$ just as good as S.

Simplified BLISS

- Secret key $S = (s_1, s_2) = (f, 2g + 1) \in R_q^2$, f, g sparse in $\{0, \pm 1\}^n$.
- Public key $A = (a_1, a_2) \in R_{2q}^2$, with key equation $a_1 s_1 + a_2 s_2 \equiv q \bmod 2q$.
- Computed as $a_q = (2g + 1)/f \bmod q$ (restart if f is not invertible); then $A = (2a_q, q - 2) \bmod 2q$.
- Can verify key guess for f with key equation; g computable.
- Focus on s_1; $-S$ just as good as S.
- To sign, sample y from discrete n-dim Gaussian $D_{\mathbf{Z}^n,\sigma}$.
- $c = H(a_1, y, \text{public stuff})$ // H hashes to R_q, sparse
- choose a random bit b.
- Signature: (z, c) with $z = y + (-1)^b s_1 \cdot c \bmod 2q$.
- Can get $\pm s_1 = (z - y)/c \in R_q$ if we know y, the error vector/polynomial; (c needs to be invertible).

Use partial information on y to attack

- Rename s_1 to s. $z = y + (-1)^b s \cdot c \bmod 2q$.
- SCA might give us only one coefficient of y per signature. How to combine?

Use partial information on y to attack

- Rename s_1 to s. $z = y + (-1)^b s \cdot c \bmod 2q$.
- SCA might give us only one coefficient of y per signature. How to combine?
- Note that z_i corresponds to coefficient of x^i.
- Then $z_i = y_i + (-1)^b \left(\sum_{j=0}^{i} s_j c_{i-j} - \sum_{j=i+1}^{n-1} s_j c_{n+i-j} \right)$.
- Have z from (z, c). Write $sc = sC, C \in \{0, \pm 1\}^{n \times n}$. Then C has known columns $c_0 = c, c_1 = xc, \ldots, c_{n-1} = x^{n-1}c$ and $z_i - y_i = (-1)^b \langle s, c_i \rangle$.
- Collect many such relations.
- No need to go for unique i; each equation involves all s_j.
- Build system of equations for the s_j.

System of equations

- Build system of equations; c, y, and z vary, s is fixed.
- Let z_i and y_i come from the i-th sample.
  Let $\mathbf{c}_i$ be the matching vector/polynomial c multiplied by the correct power of x.

$$\begin{pmatrix} (-1)^{b_0}(z_0 - y_0) \\ (-1)^{b_1}(z_1 - y_1) \\ \dots \\ (-1)^{b_{n-1}}(z_{n-1} - y_{n-1}) \end{pmatrix} = \begin{pmatrix} \mathbf{c}_0 \\ \mathbf{c}_1 \\ \vdots \\ \mathbf{c}_{n-1} \end{pmatrix} \begin{pmatrix} s_{n-1} \\ s_{n-2} \\ \vdots \\ s_0 \end{pmatrix}$$

System of equations

- Build system of equations; c, y, and z vary, s is fixed.
- Let z_i and y_i come from the i-th sample.
  Let $\mathbf{c}_i$ be the matching vector/polynomial c multiplied by the correct power of x.

$$\begin{pmatrix} (-1)^{b_0}(z_0 - y_0) \\ (-1)^{b_1}(z_1 - y_1) \\ \dots \\ (-1)^{b_{n-1}}(z_{n-1} - y_{n-1}) \end{pmatrix} = \begin{pmatrix} \mathbf{c}_0 \\ \mathbf{c}_1 \\ \vdots \\ \mathbf{c}_{n-1} \end{pmatrix} \begin{pmatrix} s_{n-1} \\ s_{n-2} \\ \vdots \\ s_0 \end{pmatrix}$$

- Looks powerful, but exhaustive search through all b_i takes 2^n (unless side-channel is very strong).

System of equations

- Build system of equations; c, y, and z vary, s is fixed.
- Let z_i and y_i come from the i-th sample.
  Let $\mathbf{c}_i$ be the matching vector/polynomial c multiplied by the correct power of x.

$$\begin{pmatrix} (-1)^{b_0}(z_0 - y_0) \\ (-1)^{b_1}(z_1 - y_1) \\ \dots \\ (-1)^{b_{n-1}}(z_{n-1} - y_{n-1}) \end{pmatrix} = \begin{pmatrix} \mathbf{c}_0 \\ \mathbf{c}_1 \\ \vdots \\ \mathbf{c}_{n-1} \end{pmatrix} \begin{pmatrix} s_{n-1} \\ s_{n-2} \\ \vdots \\ s_0 \end{pmatrix}$$

- Looks powerful, but exhaustive search through all b_i takes 2^n (unless side-channel is very strong).
- Splurge: use only entries where $z_i = y_i$, this removes b's.
  Know z_i, match with measurement y_i.
- Then solve for s (and deal with inaccurate measurements).

Results

- Analysis of CDT sampling with guide table and of rejection (Bernoulli) sampling. We get useful side-channel information.
- Interesting and hard-to-avoid leakage.
- Different precision of information, but enough from cache-timing attacks.
- Above strategy with $z_i = y_i$ in a very small set (strong side channel, but somewhat rare) takes on average 1671 signatures (for BLISS-I with $n = 512$)
- For CDT sampling always have some with uncertainty in y_i; but use much more common events.
- Use LLL to deal with uncertainty and just 441 signatures to attack CDT version,
- Run (well synchronized) spy process on same device; sample and break; actually get expected data.
- More uncertainty, more LLL (for both, CDT and Bernoulli) but works! Need about 100 extra equations for BLISS-I.

Further resources

- https://2017.pqcrypto.org/school: PQCRYPTO summer school with 21 lectures on video + slides + exercises.
- https://2017.pqcrypto.org/exec Executive school (12 lectures), less math, more overview. So far slides, soon videos.
- https://2017.pqcrypto.org/conference PQCrypto 2017; the latest results on post-quantum crypto.
- https://pqcrypto.org: Our survey site.
  - Many pointers: e.g., to PQCrypto conferences;
  - Bibliography for 4 major PQC systems.
- https://pqcrypto.eu.org: PQCRYPTO EU project. Coming soon:
  - Expert recommendations.
  - Free software libraries.
  - More benchmarking to compare cryptosystems.
- https://twitter.com/pqc_eu: PQCRYPTO Twitter feed.
- https://twitter.com/PQCryptoConf PQCrypto conference twitter feed.