



Security level analysis of AES, Trivium, Keccak and many of the
lattice-based post-quantum cryptosystems

For the mathematicians in the audience

Quantum computing generally and the theory of SVD and $k(A)$ is just finite-dimensional Linear Algebra.

But with quantum mechanics and quantum field theory, we enter the realm of Hilbert Spaces, Operator Algebras up to Quantum Geometry and Quantum Cohomology, see for instance this reference : Quantum geometric Langlands correspondence in positive characteristic :

https://www.academia.edu/1335083/Quantum_geometric_Langlands_correspondence_in_positive_characteristic_the_GL_N_case?auto=download&email_work_card=download-paper,

and eg. in quantum simulations of Fock space systems, conformal field theory is used and we are facing the above theories.



Overview of the quantum algorithm QAA by Chen-Gao 1

The QAA is a cryptographic attack on AES, Trivium and Keccak, and these cryptosystems are secure under the QAA attack only if the **condition numbers** of their corresponding equation systems are large.

This means we have a new **criterion** for the design of cryptosystems that can resist this QC attack:

Their corresponding equation systems must have large condition numbers.



Illustration of the QAA by the running example in the paper 1

As a typical example consider

$$F = \{f_1, f_2, f_3\}, X = \{x_1, x_2\}, \text{ where}$$

$$f_1 = x_1^2 - x_2, f_2 = x_2 - 1, f_3 = x_1x_2 - 1$$

We shall be looking for **Boolean solutions** over the complex numbers, ie. from $\{0,1\}$.
That means we can substitute all powers of the variables

$$x_i^2 \longmapsto x_i \in F_{bool} \quad \text{below.}$$

Note: This same procedure is applicable in so-called **QUBO** (Quadratic Unconstrained Binary Optimization) problems. These are in the class of problems amenable to the **D-Wave Quantum Annealer**.

<https://www.dwavesys.com/>



Overview of the quantum algorithm QAA by Chen-Gao 2

This quantum algorithm has 3 parts:

1. If $F \subset C[X = (x_1, \dots, x_n)]$ has the solutions a_1, \dots, a_w , a **pseudo-solution** of F is by definition a linear combination of **monomial solutions** of F :

$$\text{pseudosolution} := \sum_{c_i} m(a_i), \text{ i.e. } x_i = a_i,$$

where m is a vector of monomials in X up to a certain degree and c_i are complex numbers. It is shown that a pseudo-solution of can be computed by solving the **Macaulay Linear System** with the **HHL quantum algorithm**. The HHL gives **exponential speed-up** for solving **sparse** linear systems which is inherited by the QAA.

2. Then the authors show that **Boolean** solutions of F , i.e. from $\{0, 1\}$, can be derived from the pseudo-solutions in part 1. with high probability. The **quantum complexity**, i.e. the number of quantum gates needed to solve the problem, is polynomial in the input size and the so-called **condition number** of F .
3. Finally, part 3. is a reduction of the problem of solving a Boolean equation system, i.e. over \mathbb{Z}_2 , like that for AES, to that of the computation of the Boolean solutions for a 6-sparse polynomial system over C . We define a polynomial system P to be k -sparse iff every polynomial in P contains at most k terms.



Overview of the Quantum Algorithm QAA by Chen-Gao 3

The Macaulay Linear System

The idea of encoding a non-linear polynomial equation system as a linear system

$$M m = b,$$

where M is the **Macaulay matrix** and the solution vector m has as components possibly non-linear monomials,

dates back to the classic 1902 paper by F.S. Macaulay [Mac].



The Macaulay Matrix of the Example

$$M_{mac} = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

(f_1)

(f_2)
 $(x_2 f_2)$
 $(x_1 f_2)$

(f_3)



The Example 2

If we call the modified quantum HHL algorithm to solve this linear system, the solution vector is

$$m = (x_2, x_2^2, x_2^3, x_1, x_1 x_2, \dots) \in \mathbb{C}[X]^{15}$$

$$\text{position} = \quad 1 \quad \quad 4$$

Now, the solution vector to the polynomial system F above is

$$x = (x_2, x_1) \in C[X]^2, \text{ and } a = (x_2 = 1, x_1 = 1)$$

So we are faced with the Problem:

Let $|u\rangle$ be an N -dimensional quantum state and $n \ll N$. How can we measure n selected coordinates of $|u\rangle$ efficiently?

In the example we need to project to the 1. and 4. coordinate, ie.

$$\Pi : C[X]^{15} \longrightarrow C[X]^2$$



The Example 3

The above problem is solved by their key lemma 4.4 , which exploits one of the **key features of quantum computing** : the properties of the **quantum measurement process**.

It is proved that the quantum measurement of the monomial vector m yields a solution vector x :

$$\text{Projection } \Pi : m = (x_2, x_2^2, x_2^3, x_1, x_1x_2, \dots) \longmapsto x = (x_2, x_1)$$

Quantum Measurement

Let $|m(D)\rangle = (m(D, k))_k$ have the components $m(D, k)$.

Then the measurement of the state $|m(D)\rangle$ returns a basis vector $|e(k-1)\rangle$ in the computational basis with probability

$$p(|e(k-1)\rangle) = |m(D, k)|^2 \quad (\text{Born rule}).$$



Central Quantum Algorithm of the QAA

... implementing parts 1. and 2. above.

Input: $F = \{f_1, \dots, f_r\} \subset C[X]$. Also $\epsilon \in (0, 1)$.

Output: A Boolean solution $a \in V(F, H_X)$ (the solution variety) or $\{\}$ meaning that $V(F, H_X) = \{\}$, with success probability at least $1 - \epsilon$.

Step1 : If $F(0)=0$, return 0. If $F(1) = 0$, return 1. Set $l=1$.

Step2 : Let $F_1 := F_{\text{Bool}}$ and $Y := X$.

Step3 : Let $F_2 := F_1 \cup H_Y$ and $D := 3 \# Y$

Step4 : Use the modified HHL algorithm to solve the Macaulay linear system $M(F_2, D) m(D) = b(F_2, D)$, to obtain a state $|m(D)\rangle$ with the error bound $(\epsilon_1/n)^{0.5}$, where ϵ_1 can be chosen arbitrarily such as $\epsilon_1 = \frac{1}{2}$.

Step5 : Measuring $|m(D)\rangle$ we obtain a state $|e(k-1)\rangle$ which corresponds to $m(D, k)$ in $m(D)$.

Step6 : Let $m(D, k) = \prod x_i^{n_i}$. Set $x_i := 1$ in $F_1 \subset C[Y]$ for $i=1, \dots, u_k$.

Step7 : Remove 0 from, F_1 . Set $Y := Y \setminus \{x_i : i=1, \dots, u_k\}$.

Step8 : If $1 \in F_1$ or $Y = \{\}$ then GOTO **Step11**.

Step9 : If $F_1 \neq \{\}$ and $F_1(0) \neq 0$ then GOTO **Step3**.

Step10 : Return (a_1, \dots, a_n) where $a_i = 0$ if $x_i \in Y$, else $a_i = 1$.

Step11 : If $l > \log(\epsilon)$ then return $\{\}$, else $l := l+1$ and GOTO **Step2**.

Central Algorithm



Large Matrices, Singular Value Decomposition, Condition Numbers ...

For estimating the security levels of all of the (post-quantum) cryptosystems in question, we saw above that we have to estimate the **condition number** of a matrix A .

The following slides give a quick introduction to the necessary theory.



Algorithm: (SVD) Singular Value Decomposition

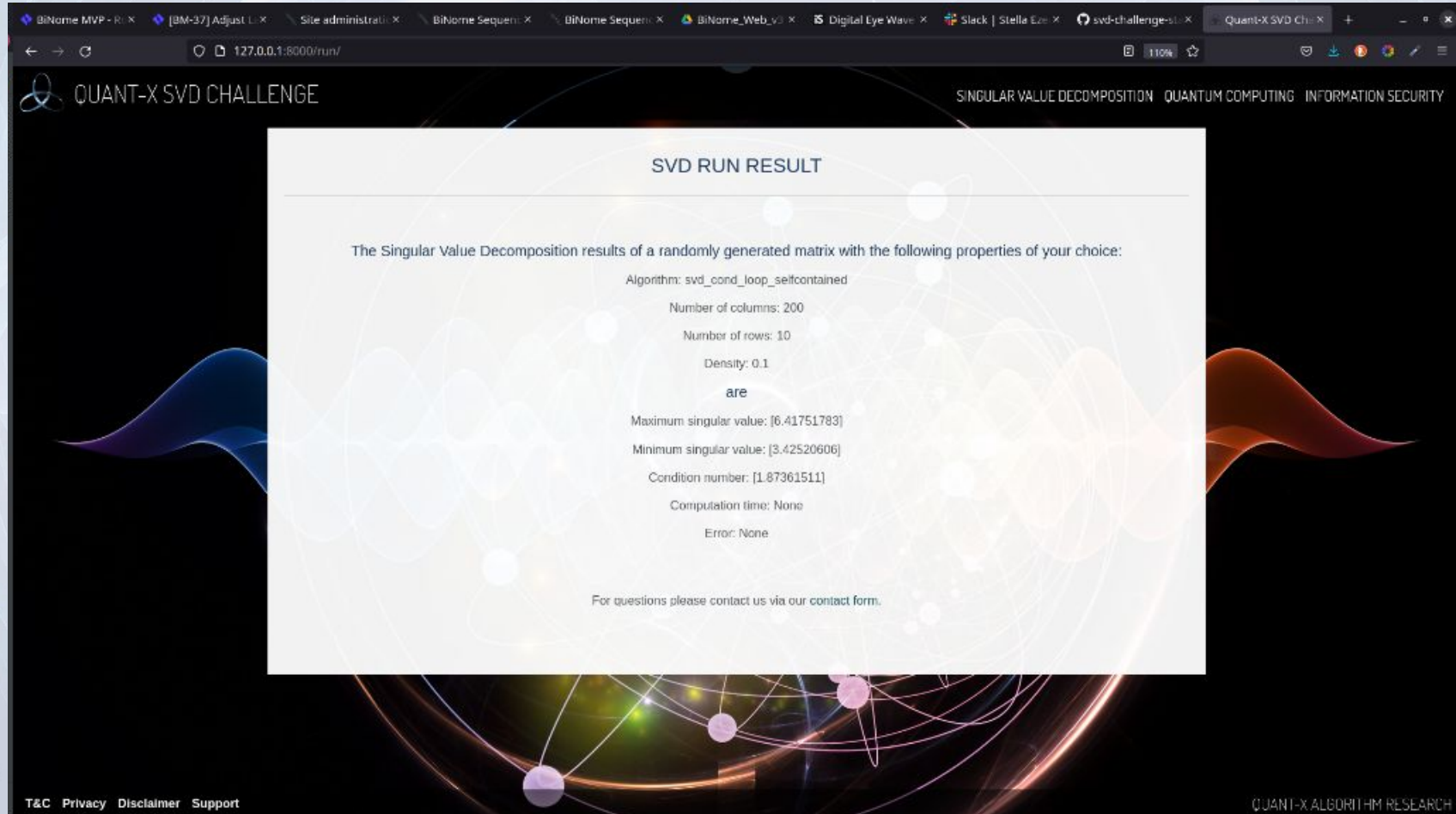
Input : An arbitrary $(m \times n)$ - matrix A over C

Output : $A = U S V^*$

1. Compute the eigenvalues λ_j of the Hermitian matrix A^*A . The singular values s_j are the square roots of the λ_j .
Fact: The number r of singular values greater than 0 is the rank of A .
2. Compute an ON Eigenbasis of A^*A . These are the right singular vectors, the columns of V .
3. The first r left singular vectors are $u_j = (1/s_j) A v_j$. The remaining left singular vectors are an ON basis of $\ker(A^*)$.



Algorithm: (SVD) Singular Value Decomposition



The screenshot shows a web browser window with multiple tabs. The active tab is 'Quant-X SVD Challenge'. The browser address bar shows '127.0.0.1:8000/run/'. The page title is 'QUANT-X SVD CHALLENGE'. The main content area is titled 'SVD RUN RESULT' and contains the following text:

The Singular Value Decomposition results of a randomly generated matrix with the following properties of your choice:

- Algorithm: `svd_cond_loop_selfcontained`
- Number of columns: 200
- Number of rows: 10
- Density: 0.1
- are
- Maximum singular value: [6.41751783]
- Minimum singular value: [3.42520606]
- Condition number: [1.87361511]
- Computation time: None
- Error: None

For questions please contact us via our contact form.

The footer of the page contains 'T&C Privacy Disclaimer Support' on the left and 'QUANT-X ALGORITHM RESEARCH' on the right, accompanied by a logo.

Definition: The Tensor Product of Matrices

The important property for numerical computations is the fact that the tensor product blows up the size of vectors and matrices exponentially.

Motto: Quantum computing often is exponentially faster than binary computing-- but the prize one has to pay is the creation of exponentially large matrices.

Reason: The n -qubit space C^{2^n} is the n -fold tensor product of the 1-qubit space C^2 .

Definition: The **tensor product** of matrices A and B is defined by

$$\otimes : C^{m \times m'} \times C^{n \times n'} \longrightarrow C^{mn \times m'n'}$$

$$(A \otimes B)(j; k) := A(j/n; k/m)B(j \bmod n; k \bmod m)$$



Tensor Product / Dirac Form of SVD

(used in quantum computing literature)

$|v\rangle := v \in C^p$ (ket - column vector).

$\langle v| := v^T$ (bra - row vector).

$|u\rangle\langle v| := uv^T$ (matrix).

$A = s_1 |u_1\rangle\langle v_1| + s_2 |u_2\rangle\langle v_2| + \dots + s_r |u_r\rangle\langle v_r|$

where $r = \text{Rank}(A)$



Definition: Condition Number κ

Let $s_1 \geq s_2 \geq \dots \geq s_r > 0$ be the singular values in descending order. Then

$$\kappa(A) := s_1(A)/s_r(A)$$

Remark:

Condition numbers of large matrices are generally very difficult even to estimate. Chen-Gao write: “ Condition numbers of equation systems are generally difficult to estimate, and estimating the condition numbers for these cryptosystems is an interesting future work ” ([CG], page 3), see the outlook on the last slide.



General Structure of a Compact Operator

Let H_1, H_2 be countably infinite-dimensional Hilbert spaces and $K(H_1, H_2)$ be the compact operators between the Hilbert spaces.

Theorem

For every $T \in K(H_1, H_2)$ exist ON-Systems $(e_i), (f_j)$ of H_1, H_2 , respectively, and a sequence of positive real numbers (s_k) , with $s_1 \geq s_2 \geq \dots \geq 0$, $\lim s_k = 0$ such that $Tx = \sum s_k \langle x | e_k \rangle f_k \in H_2$ for all $x \in H_1$.

This leads to the **informal conjecture** :

Let the dimension of the vector spaces go to infinity, i.e. the size of the operators aka matrices tend to infinity as well. Then the condition number goes to infinity for every matrix with “sufficiently high” sparsity, i.e. “sufficiently many” zero entries.



Follow-up algorithm to the QAA for Polynomial System Solving over **Finite Fields** ,
Optimization and applications to Cryptanalysis.

They reduce all of the above problems to the one solved by the QAA, i.e. this extension does not use quantum computing. As we saw above, the QAA is a quantum algorithm that finds Boolean solutions of a polynomial system over **\mathbb{C}** , the complex numbers.



Chen-Gao-Yuan II

Let F_q be a finite field , where $q = p^m$, p a prime number, $m \geq 1$. When $m = 1$, we have $F_p = 0, 1, \dots, p - 1 \bmod p$, but for $m > 1$, F_q cannot be embedded into \mathbb{Z} .

Let $P = f_1, \dots, f_r \subset F_q[X]$ be a set of polynomials in variables $X = x_1, \dots, x_n$, and $\epsilon \in (0, 1)$ be a (small) failure probability.

Theorem

There is a quantum algorithm which decides whether $P = 0$ and has a solution in F_q^n , and computes one if this is true, with success probability at least $1 - \epsilon$, and polynomial complexity in the input size, and in the condition number κ of P

This Theorem is more general than the optimization problem O stated on the next slide, because the Theorem deals with finite fields $F_q, q = p^m, m \geq 1$ whereas the optimization algorithm only works for $m = 1, F_p = 0, 1, \dots, (p - 1) \bmod p$.



Problem O by Chen-Gao-Yuan: optimization

Chen-Gao-Yuan also gives a quantum algorithm to solve the following

optimization problem over F_p

$$\min O(X, Y) \text{ subject to } X \in (F_p)^n, Y \in \mathbb{Z}^m$$

$$f_j(X) = 0 \bmod p, j = 1, \dots, r;$$

$$0 \leq g_i(X, Y) \leq b_i, i = 1, \dots, s; 0 \leq y_k \leq u_k, k = 1, \dots, m,$$

where

$$P = \{f_1, \dots, f_r\} \subset F_p[X], Y = \{y_1, \dots, y_m\}, G_O = \{O, g_1, \dots, g_s\} \subset \mathbb{Z}[X, Y],$$

$$b_i \text{ and } u_j \in \mathbb{N}.$$



Algorithm by Chen-Gao-Yuan: Cryptanalysis & Optimization

This Algorithm for solving O by Chen-Gao-Yuan is suitable for attack on lattice-based “post-quantum” cryptosystems and solving optimization problems. The attack includes the following problems which are the bases for 23 of the 69 submissions for the call by NIST to standardize the post-quantum public-key encryption systems :

1. Polynomial systems with noise problem.
2. Short integer solution problem.
3. Shortest vector problem.
4. NTRU cryptosystem (considered resistant to Shor’s quantum algorithm).
5. Learning with errors.

Problem O includes also the $(0,1)$ - programming problem as well as the **QUBO-Problem** (Quadratic unconstrained binary optimization problem)--the problem class that can be solved by **Quantum Annealing Machines** like the one by D-Wave Systems.



Outlook: Work in progress

Currently, we are constructing a **quantum algorithm** for the **computation** of the condition numbers of arbitrary matrices.

Hence, this quantum algorithm will be capable of determining the security levels of
AES, Trivium, Keccak, MPKC,
as well as that of 23 of the 69 lattice-based post-quantum cryptosystems submitted to the NIST standardization process, within the limits of current quantum computing technology.