# Estimating the cost of generic quantum pre-image attacks on SHA-2 and SHA-3

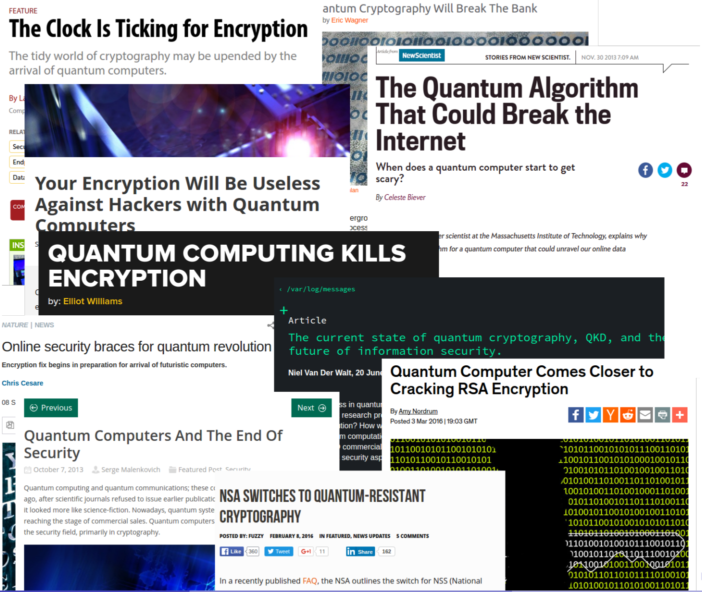
Matthew Amy Olivia Di Matteo Vlad Gheorghiu Michele Mosca
Alex Parent John Schanck

Institute for Quantum Computing, University of Waterloo

Selected Areas in Cryptography
August 12, 2016

# Introduction

Quantum computers present a threat to many asymmetric key cryptosystems



# What about other cryptosystems?

symmetric key systems (e.g. cryptographic hash functions, symmetric ciphers) weakened, but not broken.

Given a bijection

$$f : \{0,1\}^k \to \{0,1\}^k$$

a *pre-image* of $y$ is some $x$ such that $f(x) = y$. We say $f$ is *one-way* if computing a pre-image requires exhaustive search of the inputs.

Queries required to invert a $k$-bit one-way function:

| Classical | Quantum (Grover's search) |
|---|---|
| $2^k$ | $2^{k/2}$ |

$k$ bits of security $\iff$ pre-image computation takes $2^k$ queries

# How do we defend against Grover's search?

Conservative defense: double the security parameter (e.g. digest size).

Due to overhead of a realistic implementation, doubling the security may not be necessary.

*e.g. $k/2$ quantum queries may be closer to $2k/3$ classical queries*

Sources of overhead:

- Intrinsic overhead of Grover's search
- Overhead incurred at the *logical layer* by performing queries "quantumly"
- Additional overhead at the *physical layer* due to error correction

*To accurately estimate the effectiveness of a quantum attack, we need to perform a close analysis of a realistic implementation.*

# Overview

We present an estimate of the cost of performing pre-image attacks on the SHA-2 and SHA-3 families of hash functions.

A similar analysis has been performed recently for AES [M. Grassl, B. Langenberg, M. Roetteler, S. Steinwandt, "Applying Grover's algorithm to AES: quantum resource estimates", **arXiv:1512.04965**]

Methodology:

1. Implement hash function as a reversible circuit.
2. Convert to quantum gates & optimize with open-source circuit optimizer T-par.
3. Embed Grover's search in an error correcting code and analyze the physical overhead.

# Overview

**Classical query model**

Run Grovers algorithm

**Logical layer**

Generate and optimize reversible circuits

**Fault tolerant layer**

Embed reversible circuits into error correcting codes; estimate resources.

**Physical layer**

Determine physical resources (time, qubits, code cycles).

Analyzing Grover's algorithm

# Quantum computing

Classical computing:

- State of $n$ bits: $x \in \{0,1\}^n$
- Functions:
  $f : \{0,1\}^n \rightarrow \{0,1\}^m$

Quantum computing:

- State of $n$ qubits: $|\psi\rangle \in \mathbb{C}^{2^n}$
- Functions: *unitary* operators
  $U : \mathbb{C}^{2^n} \rightarrow \mathbb{C}^{2^n}$

Unitary operator = linear, invertible, norm-preserving

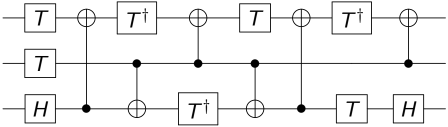
We fix a basis of $\mathbb{C}^{2^n}$ called the *computational* basis and associate each vector with a length $n$ bit-string, denoted $|x\rangle$ for $x \in \{0,1\}^n$. These are called *classical* states.

**Example**

A qubit in the state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ where $\alpha, \beta \in \mathbb{C}$ is said to be in a *superposition* of the *classical* states 0 and 1.

# Quantum circuits

Time flows left to right.




Typical gates:

$$CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \quad H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad T = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\frac{\pi}{4}} \end{pmatrix}$$

$$P = T^2, \quad Z = T^4, \quad X = HZH$$

# Oracles

Many quantum algorithms, including Grover's search, operate by applying classical functions to a superposition of states.

Problem: classical function may be irreversible

$$f(x, y) = (x, x \wedge y)$$

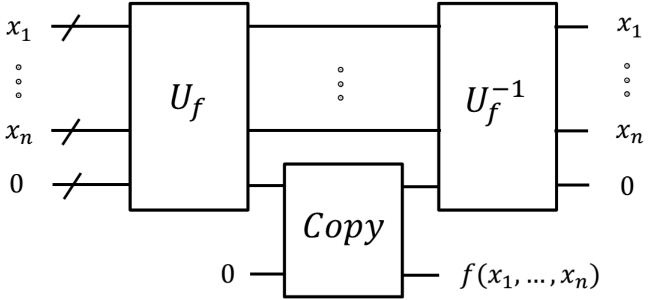
Solution: embed the function in a larger state space

$$\mathit{Toffoli}(x, y, z) = (x, y, z \oplus x \wedge y)$$

*caveat* – computations keep allocating more and more space as they run.

# The Bennett trick

Temporary space (*ancillas*) can be reclaimed by computing the function, copying output, then *uncomputing* the function.



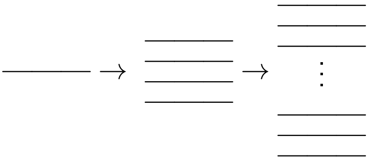

*The quantum linear systems algorithm, using Bennett's trick with oracles, inflated the number of bits from 340 to* $\sim 10^8$ *– at the logical layer!*

[A. Scherer, B. Valiron, S. Mau, S. Alexander, "Concrete resource analysis of the quantum linear system algorithm used to compute the electromagnetic scattering cross section of a 2D target", **arXiv:1505.06552**]

# Fault-tolerance

Due to short doceherence time for quantum states, some form of error correction is necessary.

To achieve fault-tolerance, *logical* qubit is encoded into many *physical* qubits via an error correcting code. This process may be iterated many times with different codes (*concatenation*) until desired error rate is achieved.



Surface code: leading modern code, simulates topological protection by "braiding" holes in a square grid around one another.
Surface code cycle: syndrome is measured and errors are corrected.

# How can we compare quantum and classical costs?

*Without significant future effort, the classical processing will almost certainly limit the speed of any quantum computer, particularly one with intrinsically fast quantum gates.* *[A. Fowler et al, "Towards practical classical processing for the surface code: Timing analysis", Phys. Rev. A* ***86****, 042313 (2012)]*

Assumptions

1. The resources required for any large quantum computation are well approximated by the resources required for that computation on a surface code based quantum computer.
2. The classical error correction routine for the surface code requires one classical processor (ASIC) per logical qubit.
3. Each ASIC performs a constant number of operations per surface code cycle.
4. The temporal cost of one surface code cycle is equal to the temporal cost of one hash function invocation

# Cost metric

## Cost metric

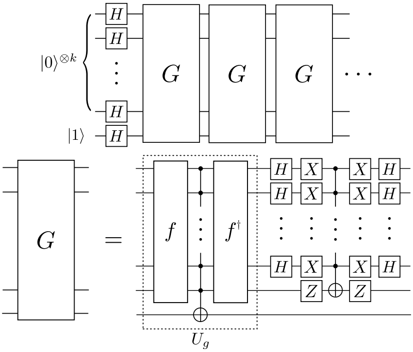
The cost of a quantum computation involving $\ell$ logical qubits for a duration of $\sigma$ surface code cycles is equal to the cost of classically evaluating a hash function $\ell \cdot \sigma$ times. Equivalently, *one logical qubit cycle* is equivalent to *one hash function invocation*.

# Analyzing Grover Part I – Grover's Algorithm

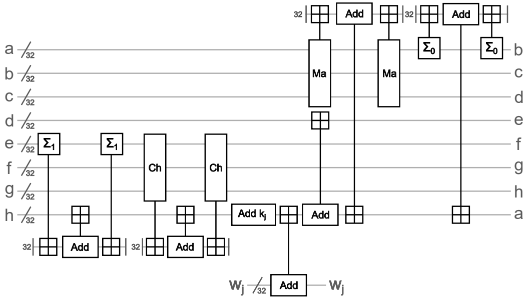
Given a predicate $g : \{0,1\}^k \rightarrow \{0,1\}$ with one solution $g(x) = 1$, Grover's search finds $x$ in $O(2^{k/2})$ queries with error $O(1/2^k)$.

Structure of Grover's search:

1. Construct superposition over all bitstrings
2. Apply Grover iterate $G$ $\lfloor \frac{\pi}{4} 2^{k/2} \rfloor$ times. $G$ uses two subroutines:
   1. $U_g$, which implements the predicate $g : x \mapsto 1$ iff $f(x) = y$
   2. The diffusion operator $2|0\rangle\langle 0| - \mathbb{I}$

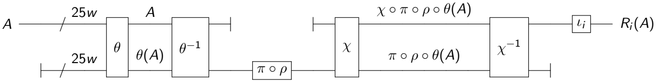


# Analyzing Grover Part II – The Oracles




SHA-256 (single round)

*In-place bits: 2402* *out-of-place bits:* $\sim$ 18000




SHA3-256 (single round)

*In-place bits: 3200* *out-of-place bits:* $\sim$ 40000

# Analyzing Grover Part III – Optimization

*Goal: reduce $T$ gates and $T$-depth (layers of parallel $T$ gates)*

| | $T$ | $P$ | $Z$ | $H$ | CNOT | $T$-Depth | Depth |
|---|---|---|---|---|---|---|---|
| SHA-256 | 401584 | 0 | 0 | 114368 | 534272 | 171552 | 528768 |
| SHA-256 (Opt.) | 228992 | 72976 | 6144 | 94144 | 4209072 | 70400 | 830720 |
| SHA3-256 | 591360 | 0 | 0 | 168960 | 33269760 | 792 | 10128 |
| SHA3-256 (Opt.) | 499200 | 46080 | 0 | 168960 | 34260480 | 432 | 11040 |

# Analyzing Grover Part IV – The Physical Layer

*Assumption: per-gate physical rates of* $p_g = 10^{-5}$.

| | | **SHA-256** | **SHA3-256** |
|---|---|---|---|
| Grover | $T$-count | $1.27 \times 10^{44}$ | $2.71 \times 10^{44}$ |
| | $T$-depth | $3.76 \times 10^{43}$ | $2.31 \times 10^{41}$ |
| | Logical qubits | 2402 | 3200 |
| | Surface code distance | 43 | 44 |
| | Physical qubits | $1.39 \times 10^{7}$ | $1.94 \times 10^{7}$ |
| $A_L$ Factories | Logical qubits per factory | 3600 | 3600 |
| | Magic state factories | 1 | 294 |
| | Surface code distances | $\{33, 13, 7\}$ | $\{33, 13, 7\}$ |
| | Physical qubits | $5.54 \times 10^{5}$ | $1.63 \times 10^{8}$ |
| Total | Logical qubits | $2^{12.6}$ | $2^{20}$ |
| | Surface code cycles | $2^{153.8}$ | $2^{146}$ |
| | Total cost | $2^{166.4}$ | $2^{166}$ |

# Summary

*Under reasonable assumptions, SHA-256 and SHA3-256 provide* 166 *bits of security against pre-image attacks in a quantum setting.*

$\Longrightarrow$ Theoretical advantages of quantum searching hide significant practical overhead!

# What's next?

- Automate & apply our scheme to other resource estimation problems.
- Find better circuit optimization techniques to reduce cost.
- Give better physical estimates by taking topological optimizations into account.
- Provide theoretical lower bounds.

Thank you!