Quantum Algorithms for Cryptanalysis and Post-Quantum Symmetric Cryptography

André Schrottenloher
Cryptography

Enable secure communications over insecure channels, at the lowest possible cost.

**Asymmetric**
- No shared secret;
- Public-key schemes (RSA...), key-exchange protocols (CSIDH...), signatures...

**Symmetric**
- *Shared secret*;
- Block ciphers (AES...), stream ciphers, hash functions (SHA-3...).
Symmetric cryptography

Alice and Bob share a secret key $k$ and communicate with a construction based on a block cipher $E_k : \{0, 1\}^n \rightarrow \{0, 1\}^n$.

Typically $n = 128$. 
Symmetric cryptography

Alice and Bob share a secret key $k$ and communicate with a construction based on a block cipher $E_k : \{0,1\}^n \rightarrow \{0,1\}^n$.

Typically $n = 128$.

An adversary tampers with the communication channel

- He can observe plaintext-ciphertext pairs $x, E_k(x)$
- He wants (for example) to recover the key
Generic attacks and cryptanalysis

The security of an **ideal** primitive is defined by **generic attacks**.

**Generic key-recovery**

- Given a few plaintext-ciphertext pairs, try all keys $k$ and find the matching one. Costs $2^{|k|}$ encryptions.
- If $|k| = 128$: $2^{128} = \text{approx. } 10^{22}$ core-years.
- “128 bits of security”

But concrete designs are not ideal and their security is **conjectural**.

- We believe that there is no better attack than generic, and so, the cipher **behaves as ideal**
- We use **cryptanalysis** as an **empirical measure of security**: if we find a better way, the cipher is **broken** (the conjecture is false)
An oversimplified overview

Start

Design a new primitive

Try to attack it

Succeed

Fail
An oversimplified overview

Ideally, we want to be stuck in **this loop**.
The adversary becomes quantum

- The **classical** security of primitives is given by a **classical** computational conjecture

Since the **quantum** adversary has a new definition of “computation”, our defenses may now be obsolete.
The post-quantum world

Asymmetric
- RSA (factorization) and ECC (discrete logarithms) become broken in polynomial time [Shor]
- Unfortunately, they are the most widely used today (replacements are on the way)

Symmetric
- Grover’s algorithm: exhaustive key-recovery becomes $\sqrt{2^{|k|}} = 2^{|k|/2}$
- Most generic attacks admit quantum replacements

$\Rightarrow$ simply “double the key size”? 

NO

---

Quantum-safe symmetric cryptography

- **Generic attacks** beyond exhaustive key search exist and **must be studied**
- We must perform **quantum cryptanalysis**

```
Start

Design a new primitive

Try to attack it

Succeed

Fail
```

post-quantum
quantumly
Quantum algorithms

- We can often treat them as abstract “black boxes”
- In cryptanalysis: **quantum search**, Simon’s algorithm, quantum walks.

Even though it’s only a picture, you’ll be able to say that I brought muffins at my thesis defense.
Quantum algorithms (feat. quantum search)

Let $X$ be a search space of size $N$, $f : X \rightarrow \{0, 1\}$, find the single $x_0 \in X$ such that $f(x) = 1$.

**Classical (exhaustive) search**

Repeat $N$ times

\[
\begin{align*}
\text{Sample } x & \in X \\
\text{Test if } f(x) & = 1
\end{align*}
\]

**Quantum search (Grover’s algorithm)**

Repeat $\mathcal{O}(\sqrt{N})$ times

\[
\begin{align*}
\text{Sample } x & \in X \rightarrow \text{quantumly} \\
\text{Test if } f(x) & = 1 \rightarrow \text{quantumly}
\end{align*}
\]

---

Grover, “A fast quantum mechanical algorithm for database search”, STOC 96
<table>
<thead>
<tr>
<th>Introduction</th>
<th>The k-XOR Problem</th>
<th>Cryptanalysis of Gimli</th>
<th>Saturnin</th>
<th>Conclusion</th>
</tr>
</thead>
</table>

## Results
Generic algorithms

André Chailloux, María Naya-Plasencia, and A. S.
An efficient quantum collision search algorithm and implications on symmetric cryptography.

Lorenzo Grassi, María Naya-Plasencia, and A. S.
Quantum algorithms for the k-XOR problem.

María Naya-Plasencia and A. S.
Optimal merging in quantum k-XOR and k-SUM algorithms.

Samuel Jaques and A. S.
Low-gate quantum golden collision finding.

Xavier Bonnetain, Rémi Bricout, A. S., and Yixin Shen.
Improved classical and quantum algorithms for subset-sum.
Dedicated cryptanalysis (symmetric)

Xavier Bonnetain, María Naya-Plasencia, and A. S.
On quantum slide attacks.

Xavier Bonnetain, Akinori Hosoyamada, María Naya-Plasencia, Yu Sasaki, and A. S.
Quantum attacks without superposition queries: The offline Simon’s algorithm.

Xavier Bonnetain, María Naya-Plasencia, and A. S.
Quantum security analysis of AES.

Patrick Derbez, Paul Huynh, Virginie Lallemand, María Naya-Plasencia, Léo Perrin, and A. S.
Cryptanalysis results on Spook - bringing full-round shadow-512 to the light.

New results on Gimli: Full-permutation distinguishers and improved collisions.
Dedicated attacks (asymmetric)


Xavier Bonnetain and A. S.
Quantum security analysis of CSIDH.
Anne Canteaut, Sébastien Duval, Gaëtan Leurent, María Naya-Plasencia, Léo Perrin, Thomas Pornin, and A. S.

Saturnin: a suite of lightweight symmetric algorithms for post-quantum security.


Ritam Bhaumik, Xavier Bonnetain, André Chailloux, Gaëtan Leurent, María Naya-Plasencia, A. S. and Yannick Seurin

QCB: Efficient Quantum-secure Authenticated Encryption

*IACR Cryptol. ePrint Arch.*, 1304 / 2020.
Contents

1 Quantum Algorithms for the k-XOR Problem
   • Collision Search
   • General k
   • With a Single Solution

2 Cryptanalysis of Gimli

3 Saturnin
Quantum Algorithms for the k-XOR Problem
**k-XOR Problem with many solutions**

**k-XOR**

Let $H : \{0, 1\}^n \rightarrow \{0, 1\}^n$ be a random function, find $x_1, \ldots, x_k$ such that $H(x_1) \oplus \ldots \oplus H(x_k) = 0$.

We suppose that quantum oracle access to $H$ is given (essentially, we can put $H$ anywhere in a quantum algorithm at cost 1).

**The query complexity**

- **Classical:** $2^{n/k}$ (trivial)
- **Quantum:** $2^{n/(k+1)}$  
  [Belovs & Spalek]

We will be interested in the time complexity, which is usually much higher.

- We focus on the exponent: $\alpha_k$ in $\tilde{O}(2^{\alpha_k n})$
- All the results apply with $+$ instead of $\oplus$

---

Belovs and Spalek, “Adversary lower bound for the k-SUM problem”, ITCS 13
The 2-XOR problem: collision search

Classical (naive): $\mathcal{O}(2^{n/2})$ computations and $\mathcal{O}(2^{n/2})$ memory.

Quantum (BHT): $\tilde{\mathcal{O}}(2^{n/3})$ computations and $\mathcal{O}(2^{n/3})$ memory.

BHT

- Store $2^{n/3}$ arbitrary queries $x, H(x)$ in a list $\mathcal{L}$
- (Grover) search $\{0, 1\}^n$ with the predicate:

$$f(x) = (\exists y \neq x, (y, H(x)) \in \mathcal{L})$$

needs $\sqrt{\frac{2^n}{2^{n/3}}} = 2^{n/3}$ iterations

---

Brassard, Høyer and Tapp, “Quantum Cryptanalysis of Hash and Claw-Free Functions”, LATIN 98
General k
Classical results

Merging

Given two lists \( \mathcal{L}_1 \) and \( \mathcal{L}_2 \), we “merge” them by taking pairs \( x_1, x_2 \in \mathcal{L}_1 \times \mathcal{L}_2 \) with a prefix condition:

\[
\mathcal{L}_1 \bowtie_s \mathcal{L}_2 = \{ x_1 \oplus x_2, x_1 \in \mathcal{L}_1, x_2 \in \mathcal{L}_2, x_1 \oplus x_2 = s \}^*
\]

All lists are presumed sorted, the time is:

\[
\max(\mid \mathcal{L}_1 \bowtie_s \mathcal{L}_2 \mid, \min(\mid \mathcal{L}_1 \mid, \mid \mathcal{L}_2 \mid))
\]

- Wagner’s algorithm consists in merging lists pairwise with arbitrary prefixes \( s \)
- The strategy depends only on \( \lfloor \log_2(k) \rfloor \); we merge \( 2^{\lfloor \log_2(k) \rfloor} \) lists
- It gives the current best time exponent: \( \mathcal{O}(2^{n/(1+\lfloor \log_2(k) \rfloor)}) \)

---

An example with $k = 4$

1. Query $2^{n/3}$ elements for each list

- $\mathcal{L}_1$ of size $2^{n/3}$
- $\mathcal{L}_2$ of size $2^{n/3}$
- $\mathcal{L}_3$ of size $2^{n/3}$
- $\mathcal{L}_4$ of size $2^{n/3}$
An example with $k = 4$

1. Query $2^{n/3}$ elements for each list
2. Merge into $\mathcal{L}_1 \bowtie_{n/3} \mathcal{L}_2$ and $\mathcal{L}_3 \bowtie_{n/3} \mathcal{L}_4$

- $\mathcal{L}_1$ of size $2^{n/3}$
- $\mathcal{L}_2$ of size $2^{n/3}$
- $\mathcal{L}_3$ of size $2^{n/3}$
- $\mathcal{L}_4$ of size $2^{n/3}$
An example with $k = 4$

1. Query $2^{n/3}$ elements for each list
2. Merge into $\mathcal{L}_1 \bowtie_{n/3} \mathcal{L}_2$ and $\mathcal{L}_3 \bowtie_{n/3} \mathcal{L}_4$ of size $2^{n/3}$
3. Merge into $(\mathcal{L}_1 \bowtie_{n/3} \mathcal{L}_2) \bowtie_{2n/3} (\mathcal{L}_3 \bowtie_{n/3} \mathcal{L}_4)$ of size 1

Single 4-XOR to 0 on $n$ bits
We search an element of $L_0$. 

Depth-first traversal of Wagner’s tree

$\mathcal{L}_1$ of size $2^{n/3}$

$\mathcal{L}_2$ of size $2^{n/3}$

$\mathcal{L}_3$ of size $2^{n/3}$

$\mathcal{L}_4$ of size $2^{n/3}$

$\mathcal{L}_1 \join_{n/3} \mathcal{L}_2$ of size $2^{n/3}$

$\mathcal{L}_3 \join_{n/3} \mathcal{L}_4$ of size $2^{n/3}$

$\mathcal{L}_0$ of size 1
We **search** an element of $L_0$.

$\implies$ We **search** an element of $L_1 \bowtie L_2$ that collides with $L_3 \bowtie L_4$.
Depth-first traversal of Wagner’s tree

We search an element of $L_0$.

$\Rightarrow$ We search an element of $L_1 \bigtriangleup L_2$ that collides with $L_3 \bigtriangleup L_4$

$\Rightarrow$ We search an element of $L_1$ that yields an element of $L_1 \bigtriangleup L_2$ that collides with $L_3 \bigtriangleup L_4$
4-XOR example

- Time $2^{n/6}$ for the search
- Time $2^{n/3}$ for the intermediate lists

---

Naya-Plasencia, S., “Optimal Merging in Quantum k-XOR and k-SUM Algorithms”, EUROCRYPT 2020
4-XOR example

- Time $2^{n/4}$ for the search
- Time $2^{n/4}$ for the intermediate lists

$\mathcal{L}_1$ of size $2^{n/2}$
$\mathcal{L}_2$ of size $2^{n/4}$
$\mathcal{L}_3$ of size $2^{n/4}$
$\mathcal{L}_4$ of size $2^{n/4}$

$\mathcal{L}_1 \bowtie_{n/4} \mathcal{L}_2$ of size $2^{n/2}$

$\mathcal{L}_3 \bowtie_{n/4} \mathcal{L}_4$ of size $2^{n/4}$

$\mathcal{L}_0$ of size 1

$\Rightarrow$ Similar results follow for all $k$

Naya-Plasencia, S., “Optimal Merging in Quantum k-XOR and k-SUM Algorithms”, EUROCRYPT 2020
General results

Exponent (with qRAM)

If $k \geq 2$ and $\kappa = \lfloor \log_2(k) \rfloor$,

$$\alpha_k = \frac{2^\kappa}{(1+\kappa)2^\kappa + k}.$$

$\implies$ the two curves have different shapes.

---

Grassi, Naya-Plasencia, S., “Quantum Algorithms for the k-XOR Problem”, ASIACRYPT 2018

Naya-Plasencia, S., “Optimal Merging in Quantum k-XOR and k-SUM Algorithms”, EUROCRYPT 2020
With a Single Solution
### The k-XOR Problem

Let $H : \{0, 1\}^{n/k} \rightarrow \{0, 1\}^n$ be a random function, find $x_1, \ldots, x_k$ such that $H(x_1) \oplus \ldots \oplus H(x_k) = 0$.

The query complexity is unchanged.

- In the classical setting, the time remains $O\left(2^{n/2}\right)$ and the memory achievable depends on $k$.
- We merge with arbitrary prefixes, and loop over these prefix choices. [Schroeppel & Shamir]
- In the quantum setting, we do the same. But the time complexity depends on $k$.

---

Schroeppel and Shamir, “A $T = O(2^{n/2}), S = O(2^{n/4})$ Algorithm for Certain NP-Complete Problems”, SIAM 1981

Naya-Plasencia, S., “Optimal Merging in Quantum k-XOR and k-SUM Algorithms”, EUROCRYPT 2020
General results

We can solve the single-solution k-XOR in time $O(2^{\gamma_k n})$ with

$$\gamma_k = \frac{k + \left\lfloor \frac{k+6}{7} \right\rfloor + \left\lfloor \frac{k+1}{7} \right\rfloor - \left\lfloor \frac{k}{7} \right\rfloor}{4k} \rightarrow \frac{2}{7} < \frac{1}{3}$$
Generic algorithms exhibit many “non-classical” behaviors:

- most gaps between $k$-XOR and $(k + 1)$-XOR exist only quantumly (the classical complexity depends only on $\lfloor \log_2(k) \rfloor$)
- Single-solution $k$-XOR ($2^{n/7} < 2^{n/3}$) goes below collision search (the classical complexities are the same)
  (single $k$-XOR is even harder due to the memory used)
<table>
<thead>
<tr>
<th>Introduction</th>
<th>The k-XOR Problem</th>
<th>Cryptanalysis of Gimli</th>
<th>Saturnin</th>
<th>Conclusion</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>Cryptanalysis of Gimli</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Cryptanalysis of Gimli**
The NIST lightweight crypto project

- Goal: standardize lightweight authenticated encryption algorithms
- 32 candidates now in the second round
- 19 of them based on permutations, with Sponge & Duplex-like modes

Gimli is a candidate based on the permutation Gimli of [Bernstein et al.].

Some of our results

- A distinguisher on the full permutation (practical on 23/24 rounds)
- **Collisions (12/24)** and semi-free start collisions (18/24) on reduced-round Gimli-Hash
- **+2 rounds** attacked in the quantum setting

---

Bernstein, Kölbl, Lucks, Massolino, Mendel, Nawaz, Schneider, Schwabe, Standaert, Todo, Viguier, “Gimli: A cross-platform permutation”, CHES 2017

Gimli is a keyless permutation operating on a 384-bit state:

$$\Pi : \{0, 1\}^{384} \rightarrow \{0, 1\}^{384}$$

The state is represented as:

- 4 "columns" $A, B, C, D$ of $96 = 3 \times 32$ bits
- each column has three 32-bit "words" $x, y, z$

It applies 24 rounds (24 to 1) of:

- **SP-Box** on each column
- (Every 2 rounds) "big" or "small" swap: swaps the $x$ words between pairs of columns
- (Every 4 rounds) **constant addition** $rc_i$
Illustration
Gimli-Hash

The 128-bit messages are injected into the \(x\) words \(A_x, B_x, C_x, D_x\).

- If the internal states collide, then the hash outputs collide
- But we control only the rate \((A_x, B_x, C_x, D_x)\)
Generic idea for collisions

Starting from a random capacity value, take a single-block message and another such that the capacity collides afterwards.

Message 1

Message 2
Idea for Gimli

- Insert a **difference in the first 32-bit word only.**
- Don’t let the difference **get away from the first column!**
- Ensure a **64-bit capacity collision at the end.**
Idea for Gimli (ctd.)

On this picture:
- **2 words of freedom** in the first column
- **3 words of freedom** coming from the sides
- **3 words of constraint** (collisions)
- **2 words of constraint** (collision)
Collision attack

**Step 1**

Find the **2 words in input** and **n words** from the sides that respect the path (approx. 1 solution).

**Step 2**

Find the **3 words of input** $B_x, C_x, D_x$ that lead to these **n words**.

Both steps can be brought down to (lots of) **double SP-Box equations**: determine the whole state given 3 **input or output wires**.

A SAT solver does that.
8-round example, Step 1

- Compute $2^{32}$ valid paths (from pairs $A_x, A'_x$)

\[
\begin{align*}
21 & \quad \Rightarrow \quad 20 \\
19 & \quad \Rightarrow \quad 18 \\
17 & \quad \Rightarrow \quad 16 \\
15 & \quad \Rightarrow \quad 14
\end{align*}
\]

\[
\text{rc}_{20} \quad \text{rc}_{16}
\]

\[
\Rightarrow \quad \text{approx. } 2^{32} \text{ double SP-Box equations}
\]
8-round example, Step 1

- Deduce $2^{32}$ valid paths (including this new word)

$\oplus$ $rc$ $\oplus$ $rc$

$\Rightarrow$ approx. $2^{32}$ double SP-Box equations
8-round example, Step 1

- Deduce $2^{32}$ valid paths (including a new word)

$ Hendra$ double SP-Box equations
8-round example, Step 1

- Find a path among $2^{32}$ that extends to a collision (including a new word)

$\oplus r_{c20}$

$\oplus r_{c16}$

$\Rightarrow$ approx. $2^{32}$ double SP-Box equations
8-round example, Step 2

From these conditions:

- Deduce this word

⇒ approx 1 double SP-Box equation
8-round example, Step 2

From **these conditions:**

- **Guess this word**, obtain the rest and check if it matches

\[ \oplus \text{rc}_{20} \]

\[ \oplus \text{rc}_{16} \]

\[ = \Rightarrow \approx 2^{32} \text{ double SP-Box equations} \]
More collision attacks

8-round collisions by solving about $2^{32}$ double SP-Box equations (practical time).

We can extend this:

- each 2 rounds add a new 32-bit condition
- 12-round collisions in $2^{96}$ equations $< 2^{256/2} = 2^{128}$

<table>
<thead>
<tr>
<th>Type</th>
<th>Rounds</th>
<th>Time (in equations)</th>
<th>Memory</th>
<th>Generic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard</td>
<td>8</td>
<td>$8 \times 2^{32}$ (practical)</td>
<td>negl.</td>
<td>$2^{128}$</td>
</tr>
<tr>
<td>Standard</td>
<td>12</td>
<td>$8 \times 2^{96}$</td>
<td>negl.</td>
<td>$2^{128}$</td>
</tr>
</tbody>
</table>
Lower margin in the quantum setting

- Classical generic bound is at $\approx 2^{256/2} = 2^{128}$ evaluations of Gimli
- Quantum generic bound is at $\approx 2^{256/3} = 2^{85.3}$ quantum evaluations of Gimli

Our collision attacks on Gimli have a square-root speedup: we can extend to 2 more rounds, like in [Hosoyamada & Sasaki]

<table>
<thead>
<tr>
<th>Type</th>
<th>Rounds</th>
<th>Time (in equations)</th>
<th>Memory</th>
<th>Generic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard</td>
<td>8</td>
<td>$8 \times 2^{32}$ (practical)</td>
<td>negl.</td>
<td>$2^{128}$</td>
</tr>
<tr>
<td>Standard</td>
<td>12</td>
<td>$8 \times 2^{96}$</td>
<td>negl.</td>
<td>$2^{128}$</td>
</tr>
<tr>
<td>Quantum</td>
<td>12</td>
<td>$\approx 8 \times 2^{48}$</td>
<td>negl.</td>
<td>$2^{85.3}$</td>
</tr>
<tr>
<td>Quantum</td>
<td>14</td>
<td>$\approx 8 \times 2^{64}$</td>
<td>negl.</td>
<td>$2^{85.3}$</td>
</tr>
</tbody>
</table>

Hosoyamada, Sasaki, ‘Finding Hash Collisions with Quantum Computers by Using Differential Trails with Smaller Probability than Birthday Bound”, EUROCRYPT 2020
Saturnin
**Context**

**Saturnin** is
- one of the 13 second-round NIST candidates based on a **block cipher**
- the only one with 256-bit blocks and (superposition) quantum security claims

<table>
<thead>
<tr>
<th>5</th>
<th>4</th>
<th>3</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Saturnin:</td>
<td>a suite of</td>
<td>lightweight</td>
<td>symmetric algorithms for</td>
<td>post-quantum security</td>
</tr>
</tbody>
</table>

1. **we wanted to build a block cipher**
2. **... post-quantum:** 256-bit keys **and blocks**, quantum security claims
3. **... lightweight:** performs well on all platforms
4. **with quantum-secure modes of operation** for AEAD / Hashing
5. **and a good-sounding name**

On the name

- As the hero of a kids TV show in the 60's, Saturnin is a (the most?) 
  famous french duck
The state

4 × 4 × 4 cube of 4-bit nibbles

Operations are easier to describe

16 registers of 16 bits

Good for implementations

16 values of 16 bits (the columns)

Looks like the state of the cipher AES
The round function

One round of Saturnin

- **S-Box layer**
- **Nibble permutation** \(SR\) and its inverse
- **Linear MixColumns**
- Every two rounds: **Sub-key addition** (and round constants)

Two rounds of Saturnin

Similar to a single round of AES in the AES-like representation.

- AES-128 has **10 rounds**: Saturnin has **20 rounds**.
- AES has very simple security arguments: Saturnin also.
- AES has 20 years of cryptanalysis: Saturnin benefits from it.
About Saturnin-CTR-Cascade

In order to obtain an authenticated cipher, we used the encrypt-then-MAC paradigm:

- encrypt with the counter mode (CTR)
- then authenticate with a Cascade MAC

This creates a quantum-secure AE at a rate of 2 encryptions per block.

Can we do the same with one encryption per block?

- Yes, with a quantum-secure **tweakable** block cipher.
- Yes, with a related-key quantum-secure block cipher.
- With a block cipher, without related-key assumptions: **open question**.

---

*Bhaumik, Bonnetain, Chailloux, Leurent, Naya-Plasencia, S., Seurin, “QCB: Efficient Quantum-secure Authenticated Encryption”*
Conclusion
Conclusion

Generic algorithms “behave” differently in the classical / quantum setting.

**Example:** collisions and 4-XOR

Some attacks work “better” in the quantum setting than classically.

**Example:** quantum collision attacks

Quantum security does not come at the expense of lightness.

**Example:** Saturnin has a large block size, but the cipher remains a contender in the “lightweight” category.
Some attacks work “better” in the quantum setting than classically.

- When time and memory can be improved simultaneously (offline-Simon)
- When the generic attack is relatively less efficient: quantum collision attacks against hash functions
- When superposition query access is allowed
Thank you!
On input $x||y||z$:

1. Rotate $x$ and $y$: $x \leftarrow x \ll 24$, $y \leftarrow y \ll 9$.

2. Perform the following non-linear operations in parallel (note that shifts are used here instead of rotations):
   
   \[ x \leftarrow x \oplus (z \ll 1) \oplus ((y \land z) \ll 2), \]
   \[ y \leftarrow y \oplus x \oplus ((x \lor z) \ll 1), \]
   \[ z \leftarrow z \oplus y \oplus ((x \land y) \ll 3). \]

3. Swap $x$ and $z$: $(x, z) \leftarrow (z, x)$. 