Quantum Algorithms for Cryptanalysis and Post-Quantum Symmetric Cryptography

André Schrottenloher

European Research Council
Established by the European Commission

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...

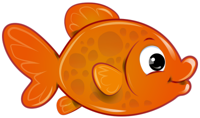
Symmetric

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

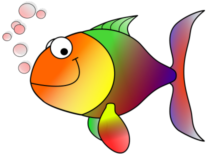
Symmetric cryptography

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

Typically $\mathbf{n} = 128$.



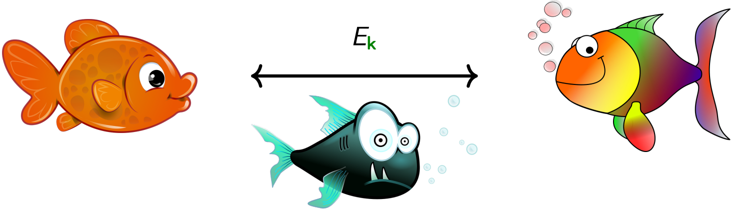
$$\xleftrightarrow{\quad E_{\mathbf{k}} \quad}$$



Symmetric cryptography

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

Typically $\mathbf{n} = 128$.



An adversary tampers with the communication channel

- He can observe plaintext-ciphertext pairs $x, E_{\mathbf{k}}(x)$
- He wants (for example) to recover the key

Generic attacks and cryptanalysis

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

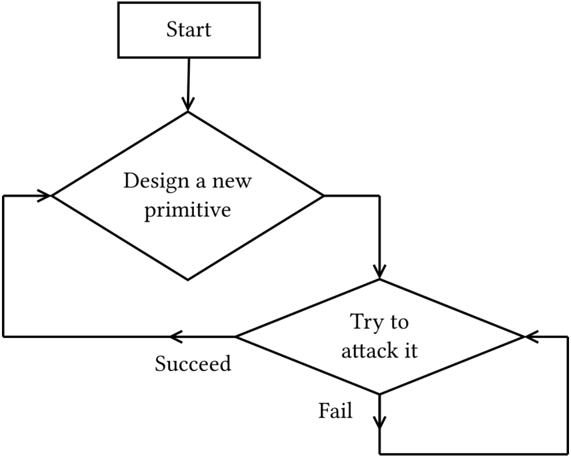
Generic key-recovery

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

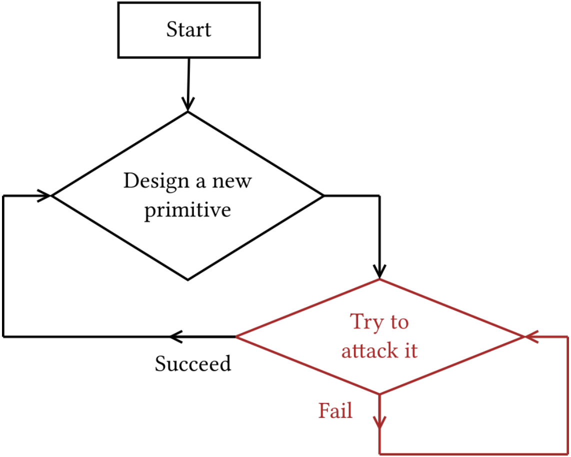
But concrete designs are not ideal and their security is **conjectural**.

- 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

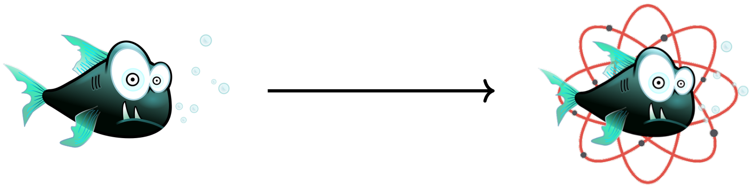


An oversimplified overview



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 $\sqrt{2^{|\mathbf{k}|}} = 2^{|\mathbf{k}|/2}$
- Most generic attacks admit quantum replacements

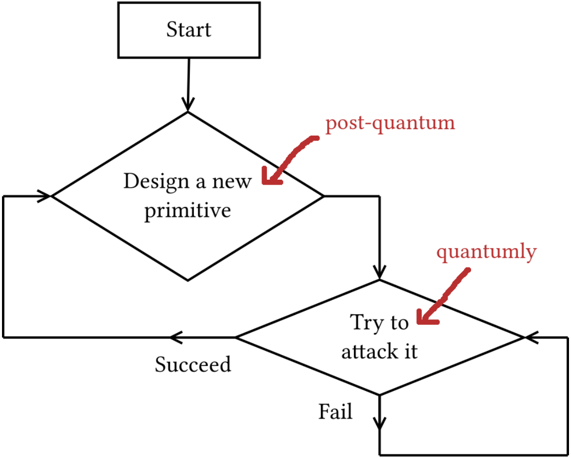
$\Longrightarrow$ simply "double the key size"?

NO

Shor, "Algorithms for Quantum Computation: Discrete Logarithms and Factoring", FOCS 1994

Quantum-safe symmetric cryptography

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



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)

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

Classical (exhaustive) search

$$\text{Repeat } N \text{ times} \begin{cases} \text{Sample } x \in X \\ \text{Test if } f(x) = 1 \end{cases}$$

Quantum search (Grover's algorithm)

$$\text{Repeat } \mathcal{O}\left(\sqrt{N}\right) \text{ times} \begin{cases} \text{Sample } x \in X \to \text{quantumly} \\ \text{Test if } f(x) = 1 \to \text{quantumly} \end{cases}$$

Grover, "A fast quantum mechanical algorithm for database search", STOC 96

Results

Generic algorithms

André Chailloux, María Naya-Plasencia, and A. S.
An efficient quantum collision search algorithm and implications on symmetric cryptography.
In *ASIACRYPT (2)*, volume 10625 of *LNCS*. Springer, 2017.

Lorenzo Grassi, María Naya-Plasencia, and A. S.
Quantum algorithms for the k-XOR problem.
In *ASIACRYPT 2018*, volume 11272 of *LNCS*. Springer, 2018.

María Naya-Plasencia and A. S.
Optimal merging in quantum k-XOR and k-SUM algorithms.
In *EUROCRYPT (2)*, volume 12106 of *LNCS*. Springer, 2020.

Samuel Jaques and A. S.
Low-gate quantum golden collision finding.
In *SAC*, LNCS. Springer, 2020.

Xavier Bonnetain, Rémi Bricout, A. S., and Yixin Shen.
Improved classical and quantum algorithms for subset-sum.
In *ASIACRYPT*, LNCS. Springer, 2020.

Dedicated cryptanalysis (symmetric)

Xavier Bonnetain, María Naya-Plasencia, and A. S.
On quantum slide attacks.
In *SAC*, volume 11959 of *LNCS*. Springer, 2019.

Xavier Bonnetain, Akinori Hosoyamada, María Naya-Plasencia, Yu Sasaki, and A. S.
Quantum attacks without superposition queries: The offline Simon's algorithm.
In *ASIACRYPT (1)*, volume 11921 of *LNCS*. Springer, 2019.

Xavier Bonnetain, María Naya-Plasencia, and A. S.
Quantum security analysis of AES.
IACR Trans. Symmetric Cryptol., 2019(2):55–93, 2019.

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.
In *CRYPTO (3)*, volume 12172 of *LNCS*, pages 359–388. Springer, 2020.

Antonio Flórez-Gutiérrez, Gaëtan Leurent, María Naya-Plasencia, Léo Perrin, A. S., and Ferdinand Sibleyras.
New results on Gimli: Full-permutation distinguishers and improved collisions.
In *ASIACRYPT 2020, LNCS*, volume 2020, page 744, 2020.

Dedicated attacks (asymmetric)

Jean-François Biasse, Xavier Bonnetain, Benjamin Pring, A. S., and William Youmans.
A trade-off between classical and quantum circuit size for an attack against CSIDH.
Journal of Mathematical Cryptology, pages 1–16, 2019.

Xavier Bonnetain and A. S.
Quantum security analysis of CSIDH.
In *EUROCRYPT (2)*, volume 12106 of *LNCS*, pages 493–522. Springer, 2020.

Design

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.
IACR Trans. Symmetric Cryptol., 2020.

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

Quantum Algorithms for the k-XOR Problem

k-XOR problem with many solutions

k-XOR

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

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: $2^{\mathbf{n}/\mathbf{k}}$ (trivial)

Quantum: $2^{\mathbf{n}/(\mathbf{k}+1)}$ [Belovs & Spalek]

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

- We focus on the exponent: $\alpha_{\mathbf{k}}$ in $\widetilde{\mathcal{O}}\left(2^{\alpha_{\mathbf{k}}\mathbf{n}}\right)$
- 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): $\mathcal{O}\left(2^{n/2}\right)$ computations and $\mathcal{O}\left(2^{n/2}\right)$ memory.

Quantum (BHT): $\widetilde{\mathcal{O}}\left(2^{n/3}\right)$ computations and $\mathcal{O}\left(2^{n/3}\right)$ 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:

$$\mathcal{L}_1 \bowtie_{\mathbf{s}} \mathcal{L}_2 = \{x_1 \oplus x_2, x_1 \in \mathcal{L}_1, x_2 \in \mathcal{L}_2, x_1 \oplus x_2 = \mathbf{s}|*\}$$

All lists are presumed sorted, the time is:

$$\mathrm{MAX}\left(|\mathcal{L}_1 \bowtie_{\mathbf{s}} \mathcal{L}_2|, \mathrm{MIN}\left(|\mathcal{L}_1|, |\mathcal{L}_2|\right)\right)$$

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

Wagner, "A Generalized Birthday Problem", CRYPTO 2002

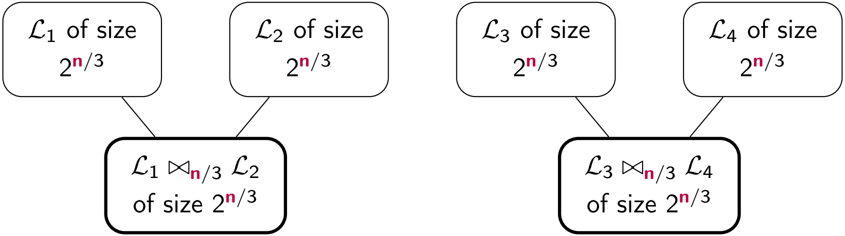
An example with $\mathbf{k} = 4$

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

$\mathcal{L}_1$ of size $2^{\mathbf{n}/3}$	$\mathcal{L}_2$ of size $2^{\mathbf{n}/3}$	$\mathcal{L}_3$ of size $2^{\mathbf{n}/3}$	$\mathcal{L}_4$ of size $2^{\mathbf{n}/3}$

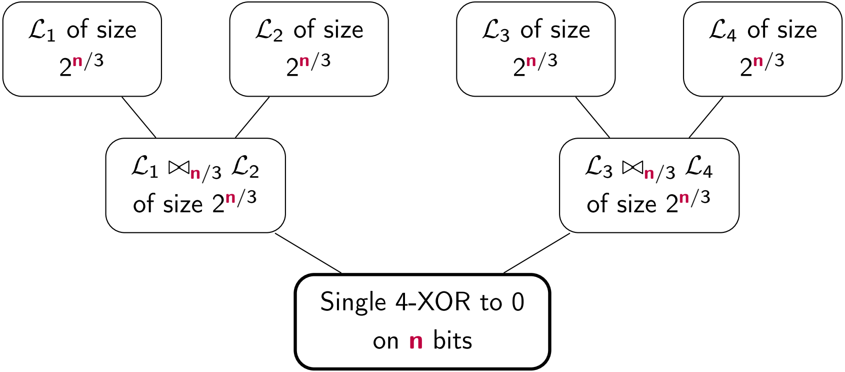
An example with $\mathbf{k} = 4$

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



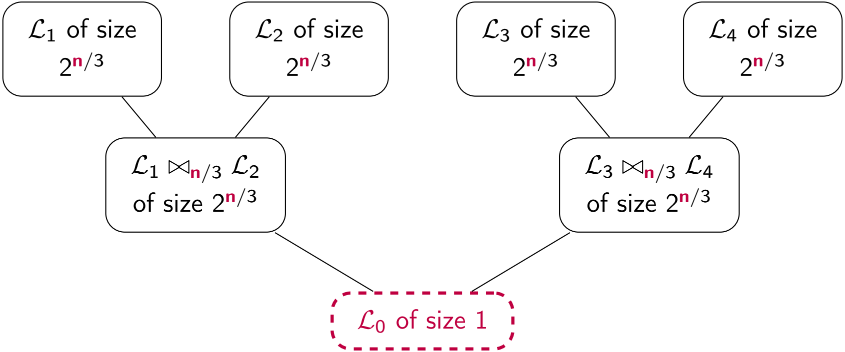
An example with $\mathbf{k} = 4$

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



Depth-first traversal of Wagner's tree

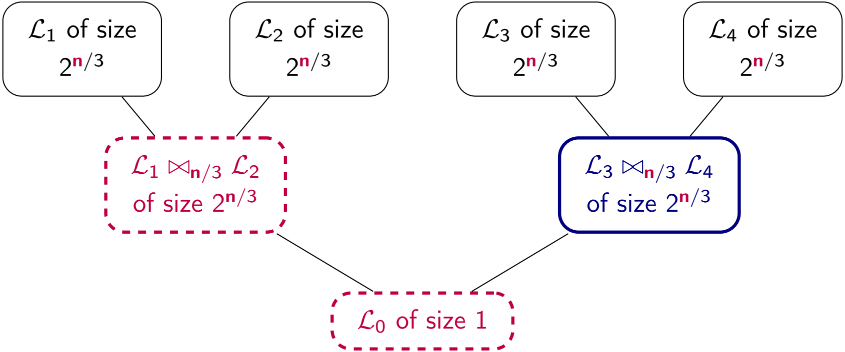
We **search** an element of $\mathcal{L}_0$.



Depth-first traversal of Wagner's tree

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

$\Longrightarrow$ We **search** an element of $\mathcal{L}_1 \bowtie \mathcal{L}_2$ that collides with $\mathcal{L}_3 \bowtie \mathcal{L}_4$

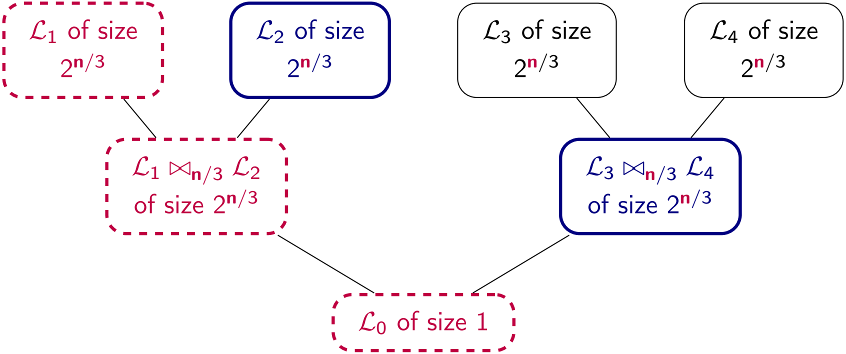


Depth-first traversal of Wagner's tree

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

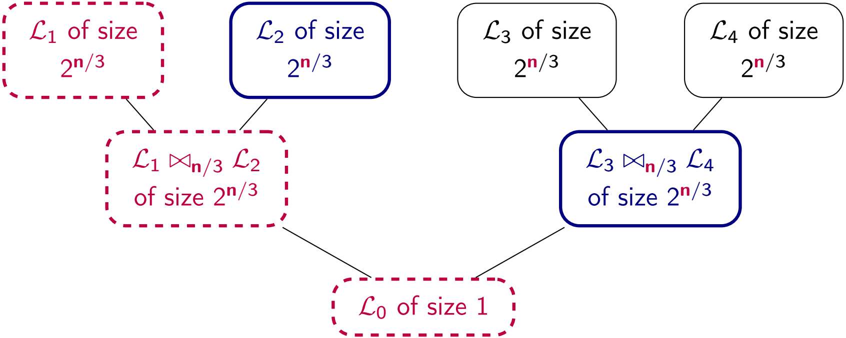
$\Longrightarrow$ We **search** an element of $\mathcal{L}_1 \bowtie \mathcal{L}_2$ that collides with $\mathcal{L}_3 \bowtie \mathcal{L}_4$

$\Longrightarrow$ We **search** an element of $\mathcal{L}_1$ that yields an element of $\mathcal{L}_1 \bowtie \mathcal{L}_2$ that collides with $\mathcal{L}_3 \bowtie \mathcal{L}_4$



4-XOR example

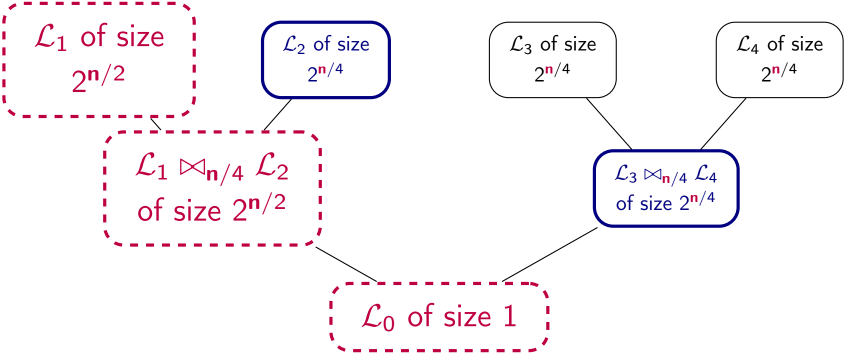
- Time $2^{n/6}$ for the **search**
- Time $2^{n/3}$ for the **intermediate lists**



Naya-Plasencia, S., "Optimal Merging in Quantum k-XOR and k-SUM Algorithms", EUROCRYPT 2020

4-XOR example

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



$\Longrightarrow$ Similar results follow for all **k**

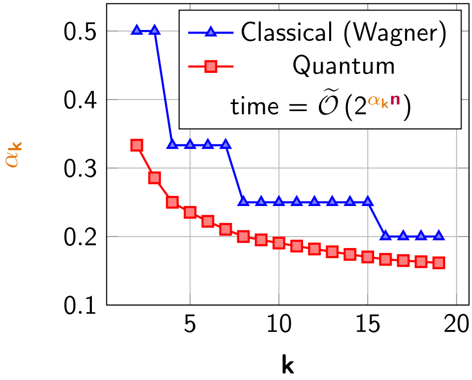
Naya-Plasencia, S., "Optimal Merging in Quantum k-XOR and k-SUM Algorithms", EUROCRYPT 2020

General results

Exponent (with qRAM)

If $\mathbf{k} \geq 2$ and $\kappa = \lfloor \log_2(\mathbf{k}) \rfloor$:
$\alpha_{\mathbf{k}} = \frac{2^{\kappa}}{(1+\kappa)2^{\kappa}+\mathbf{k}}$.

$\Longrightarrow$ 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

k-XOR problem with a single solution

k-XOR

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

The query complexity is unchanged.

- In the classical setting, the time remains $\mathcal{O}\left(2^{\mathbf{n}/2}\right)$ 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**.

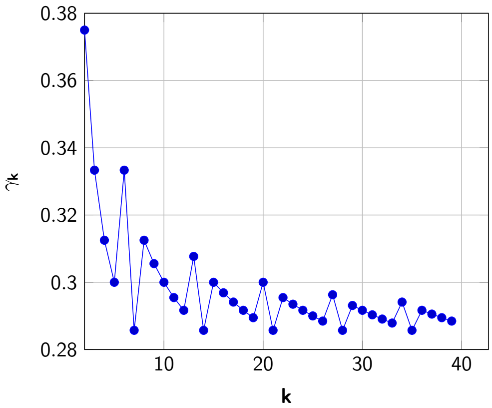
Schroeppel and Shamir, "A $T = \mathcal{O}(2^{n/2}), S = \mathcal{O}(2^{n/4})$ Algorithm for Certain NP-Complete Problems", SIAM 1981

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 $\mathcal{O}\left(2^{\gamma_{\mathbf{k}}\mathbf{n}}\right)$ with

$$\gamma_{\mathbf{k}} = \frac{\mathbf{k} + \left\lfloor \frac{\mathbf{k}+6}{7} \right\rfloor + \left\lfloor \frac{\mathbf{k}+1}{7} \right\rfloor - \left\lfloor \frac{\mathbf{k}}{7} \right\rfloor}{4\mathbf{k}} \to \frac{2}{7} < \frac{1}{3}$$



Conclusion

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 ($2^{2\mathbf{n}/7} < 2^{\mathbf{n}/3}$) goes below collision search (the classical complexities are the same) (single k-XOR is even **harder** due to the memory used)

Cryptanalysis of Gimli

Context

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

Flórez-Gutiérrez, Leurent, Naya-Plasencia, Perrin, S. and Sibleyras, "New results on Gimli: full-permutation distinguishers and improved collisions", ASIACRYPT 2020

Specification of Gimli

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

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

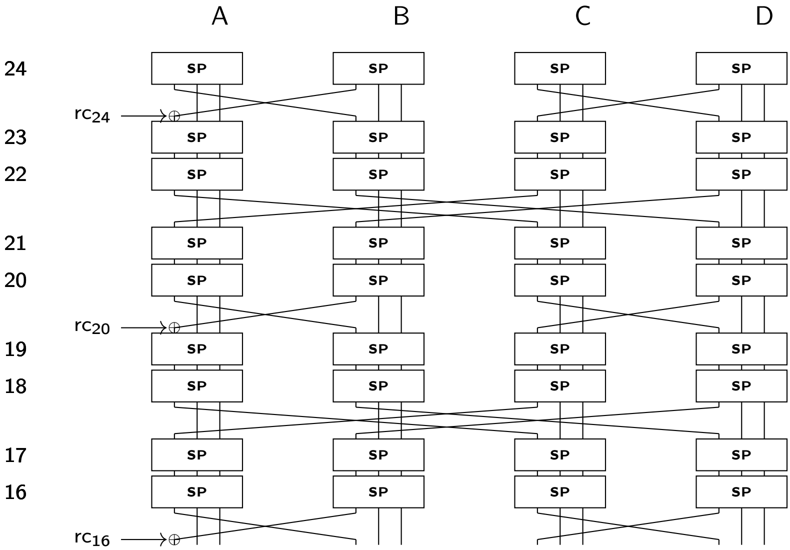
The state is represented as:

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

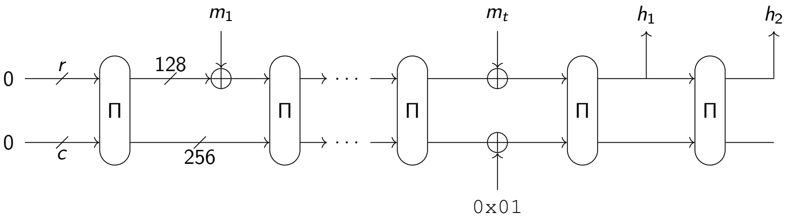
It applies **24** rounds (24 to 1) of:

- **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

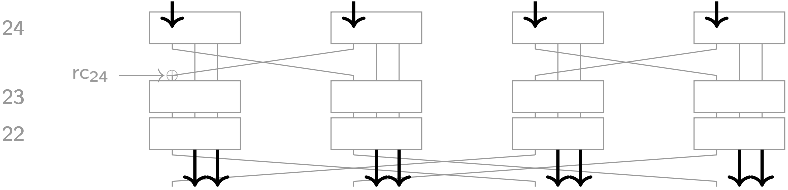
Illustration



Gimli-Hash



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

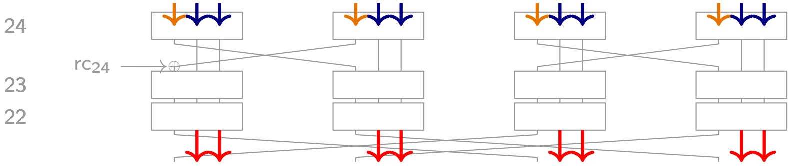


- **If** the internal states collide, then the hash outputs collide
- **But** we control only the rate (A_x, B_x, C_x, D_x)

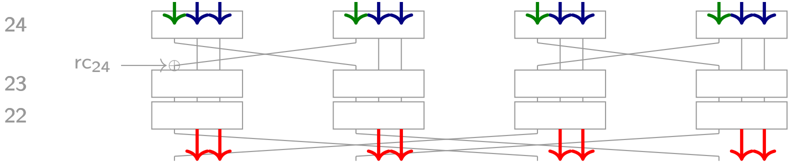
Generic idea for collisions

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

Message 1

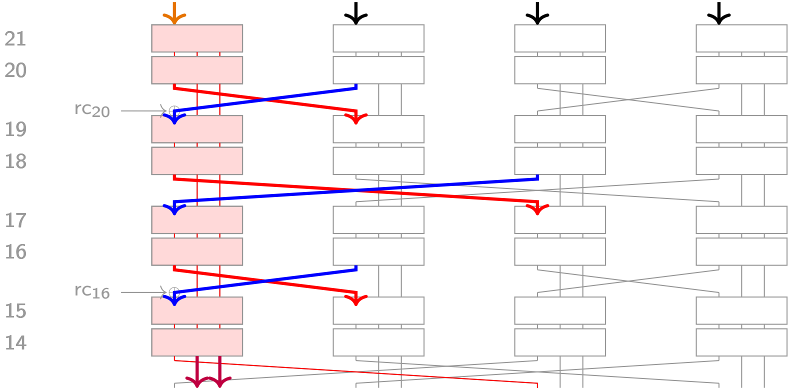


Message 2

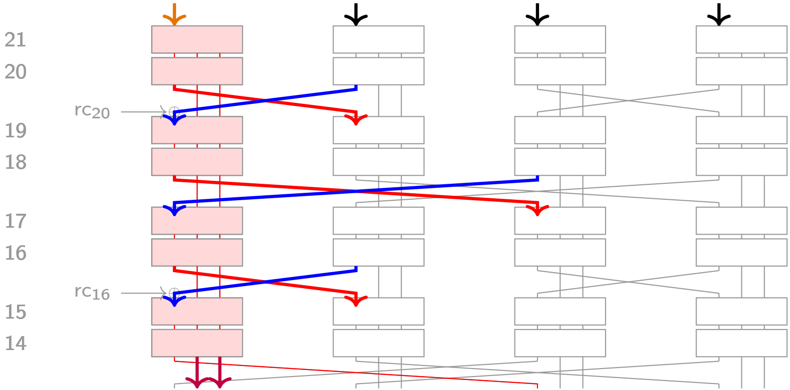


Idea for Gimli

- 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)

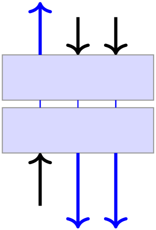
Collision attack

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**.

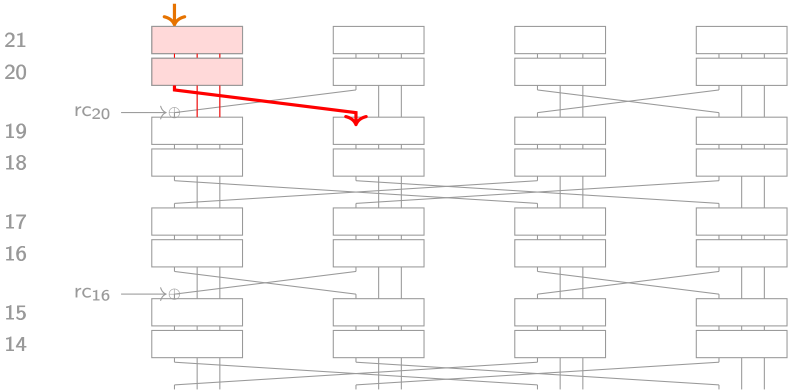


Both steps can be brought down to (lots of) **double SP-Box equations**: determine the whole state given **3 input or output wires**.

A SAT solver does that.

8-round example, Step 1

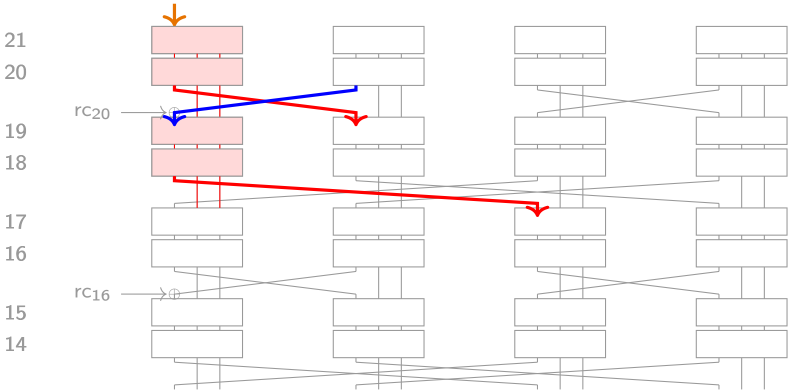
- Compute 2^{32} valid paths (from **pairs** A_x, A'_x)



$\implies$ approx. 2^{32} double SP-Box equations

8-round example, Step 1

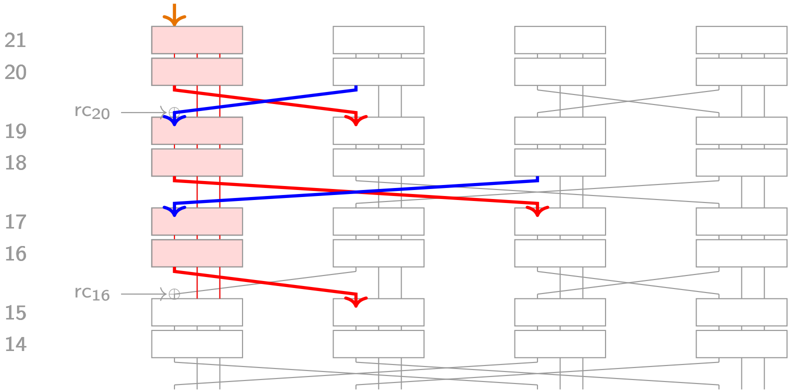
- Deduce 2^{32} valid paths (including **this new word**)



$\Longrightarrow$ approx. 2^{32} double SP-Box equations

8-round example, Step 1

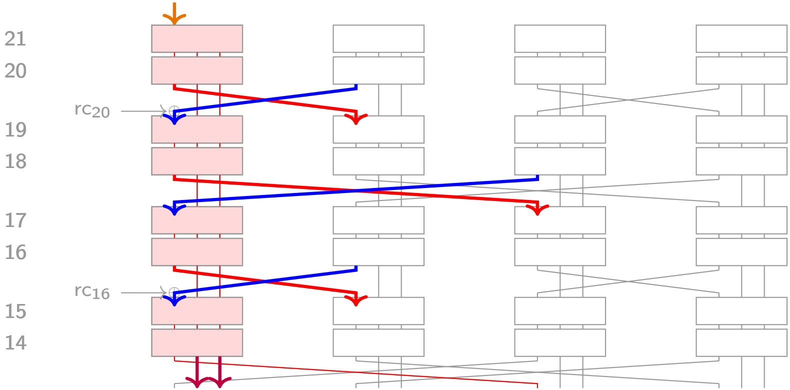
- Deduce 2^{32} valid paths (including **a new word**)



$\Longrightarrow$ approx. 2^{32} double SP-Box equations

8-round example, Step 1

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

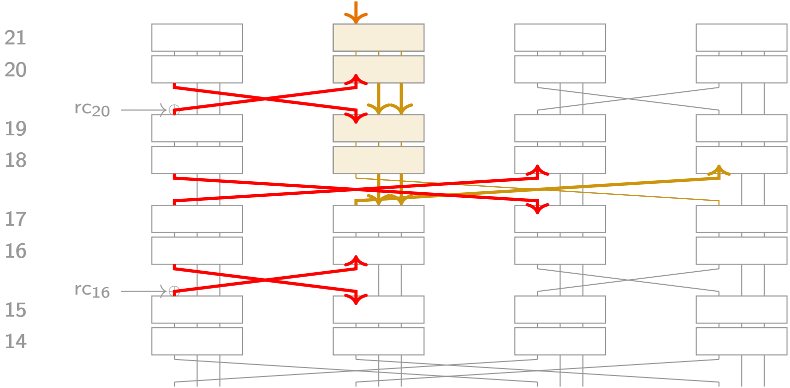


$\Longrightarrow$ approx. 2^{32} double SP-Box equations

8-round example, Step 2

From **these conditions:**

- Deduce **this word**

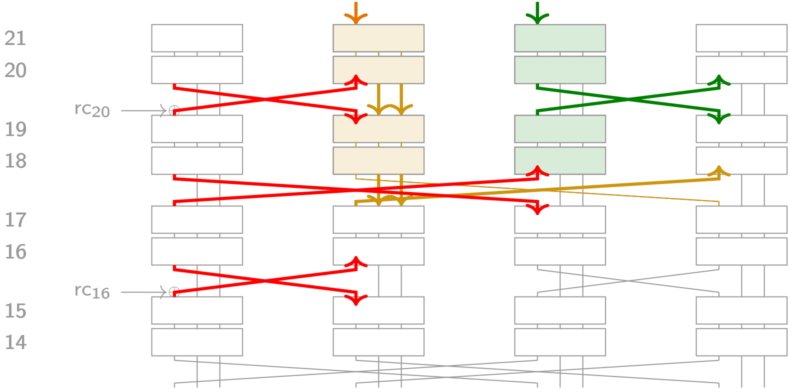


$\Longrightarrow$ approx 1 double SP-Box equation

8-round example, Step 2

From **these conditions:**

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



$\Longrightarrow$ approx 2^{32} 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 $< 2^{256/2} = 2^{128}$

Type	Rounds	Time (in equations)	Memory	Generic
Standard	8	8×2^{32} (practical)	negl.	2^{128}
Standard	12	8×2^{96}	negl.	2^{128}

Lower margin in the quantum setting

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

Type	Rounds	Time (in equations)	Memory	Generic
Standard	8	8×2^{32} (practical)	negl.	2^{128}
Standard	12	8×2^{96}	negl.	2^{128}
Quantum	12	$\simeq 8 \times 2^{48}$	negl.	$2^{85.3}$
Quantum	14	$\simeq 8 \times 2^{64}$	negl.	$2^{85.3}$

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

5	4	3	1	2
Saturnin:	a suite of	lightweight	symmetric algorithms for	post-quantum security

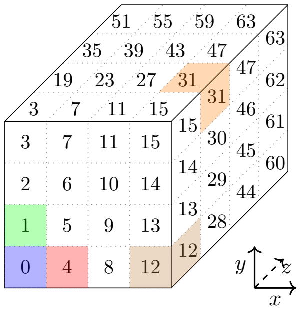
1. **we wanted to build a block cipher**
2. **. . . post-quantum:** 256-bit keys **and blocks**, quantum security claims
3. **. . . lightweight:** performs well on all platforms
4. **with quantum-secure modes of operation** for AEAD / Hashing
5. **and a good-sounding name**

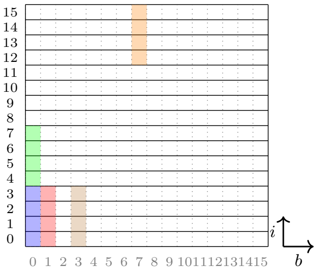
Canteaut, Duval, Leurent, Naya-Plasencia, Perrin, Pornin, S., "Saturnin: a suite of lightweight symmetric algorithms for post-quantum security", IACR Trans. Symmetric Cryptol. S1, 2020

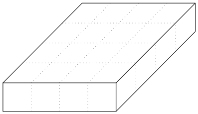
On the name

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

The state







$4 \times 4 \times 4$ cube of 4-bit nibbles

16 registers of 16 bits

16 values of 16 bits (the columns)

Operations are easier to describe

Good for implementations

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.

Conclusion (ctd.)

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!

Additional material

Gimli SP-Box

On input $x \| y \| z$:

1. Rotate x and y: $x \leftarrow x \lll 24, y \leftarrow y \lll 9$.
2. Perform the following non-linear operations in parallel (note that **shifts** are used here instead of rotations):

$$\begin{aligned} x &\leftarrow x \oplus (z \ll 1) \oplus ((y \wedge z) \ll 2) \ , \\ y &\leftarrow y \oplus x \oplus ((x \vee z) \ll 1) \ , \\ z &\leftarrow z \oplus y \oplus ((x \wedge y) \ll 3) \ . \end{aligned}$$

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