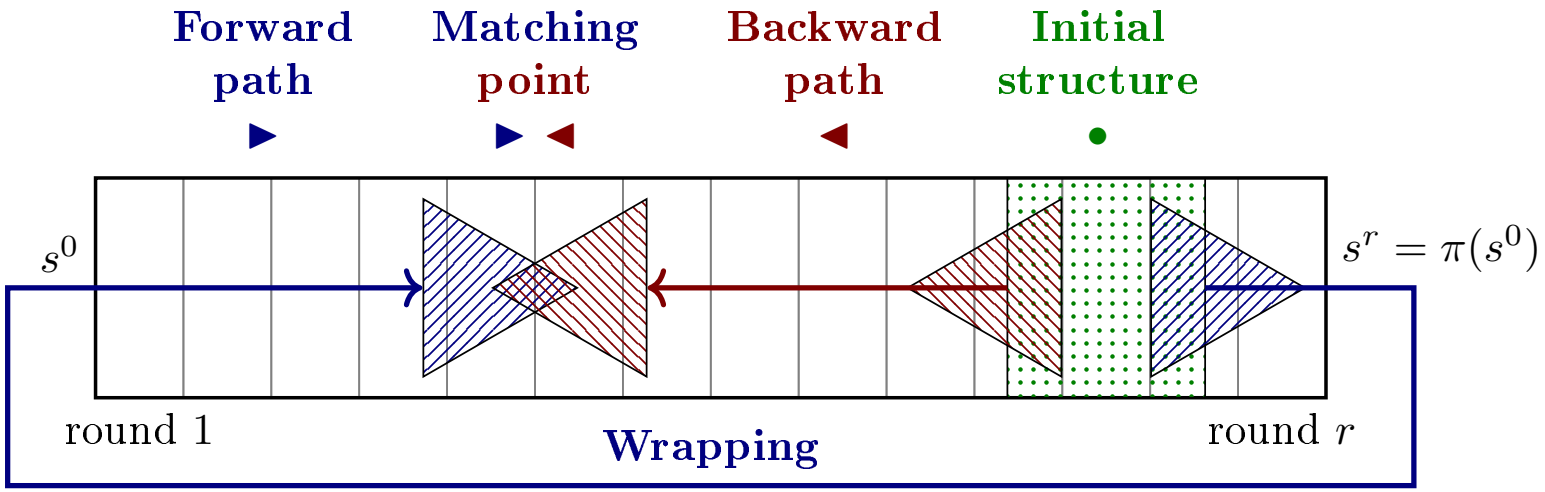
Simplified MITM Modeling for Permutations: New (Quantum) Attacks

André Schrottenloher and Marc Stevens

Cryptology Group, CWI

MITM attacks on permutations

- We search for x with some relation between x and $P(x)$: e.g. $x = P(x)$ (fixpoint)



1. Compute along a **forward computational path**
2. (Independently) compute along a **backward path**
3. Try to match the paths

Motivation: hash function preimages

Example: Haraka-512 (v2)

$$\text{Haraka} - 512(x) = \text{trunc}_{256}(x \oplus P(x))$$

defined using a **permutation** P on 512 bits.

- Finding a preimage is a **MITM problem**
  ex.: find x such that $\text{trunc}_{256}(x) = \text{trunc}_{256}(P(x))$
- Happens more generally in compression functions based on block ciphers, **when we fix the key input**


Kölbl, Lauridsen, Mendel, Rechberger, "Haraka v2 - efficient short-input hashing for post-quantum applications", ToSC 2016

Quantum MITM attacks

- Haraka has been explicitly designed for post-quantum hash-based signatures (e.g. SPHINCS+)
- We need to look at its **classical and quantum** preimage security

- Generic classical preimage in time 2^{256} (exhaust. search)
- Generic quantum preimage in time 2^{128} (Grover search)

- Haraka-512 is broken: there is a MITM preimage attack in time $2^{240} < 2^{256}$ **[BDGLSSW21]**
- But it could still be quantumly safe

- (New) Haraka-512 is **quantumly broken:** there is a **quantum** MITM preimage attack in time $< 2^{128}$


Bao, Dong, Guo, Li, Shi, Sun, Wang, "Automatic search of Meet-in-the-middle preimage attacks on AES-like hashing", EUROCRYPT 2021

Automatic search of MITM attacks

[BDGLSSW21] use a MILP modeling to search for MITM attacks:

- they target **AES-like** compression functions

> **Optimize** "the attack complexity"
>
> **among** "all possible **forward** ▶ / **backward** ◀ paths"
> (constrained by a set of **local rules**)

We use a different modeling strategy, which targets **permutations**

- No key-schedules like **[BDGLSSW21]**, but more than AES-like
- The MILP model is **extremely simple**
- Our objective includes **quantum MITM attacks**
- Applications to AES, **Haraka**, Grøstl, Spongent, Simpira, Sparkle

Outline

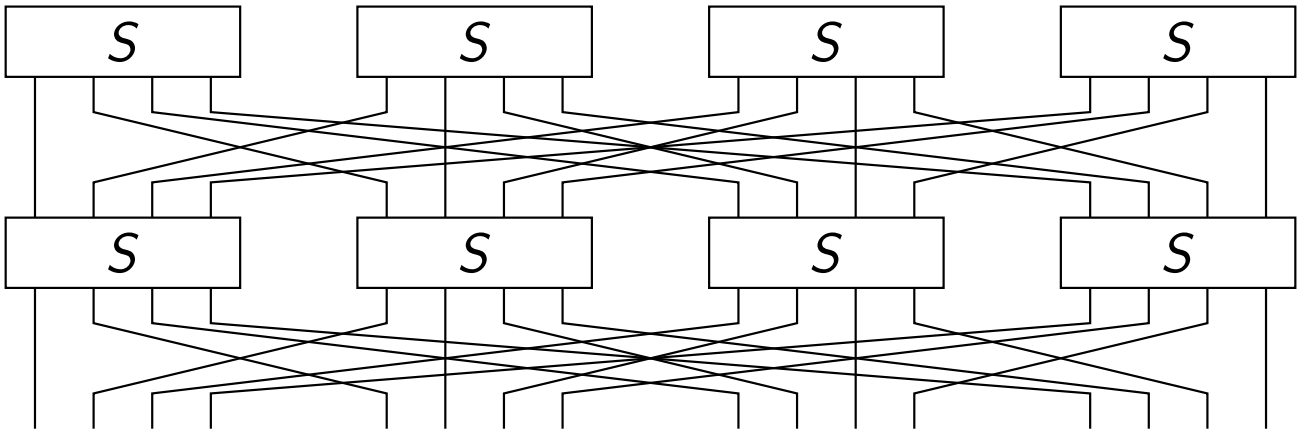
Representing a permutation

SPN permutations and Present

An SPN permutation is like an SPN cipher without a key.

- State: b **cells** of w bits each
- Round: **S-Box layer** (S) and **linear layer** (P)

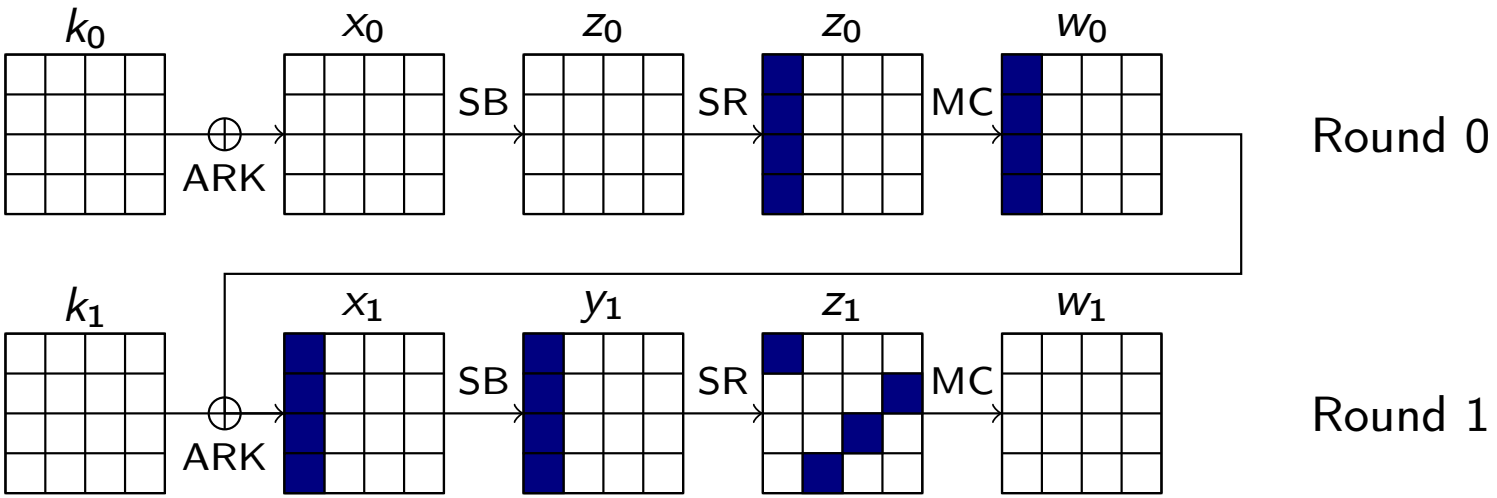
Present is a block cipher with 16 cells of 4 bits (= 64 bits). The linear layer permutes the bits.



(That's a small **Present** with 4 cells).

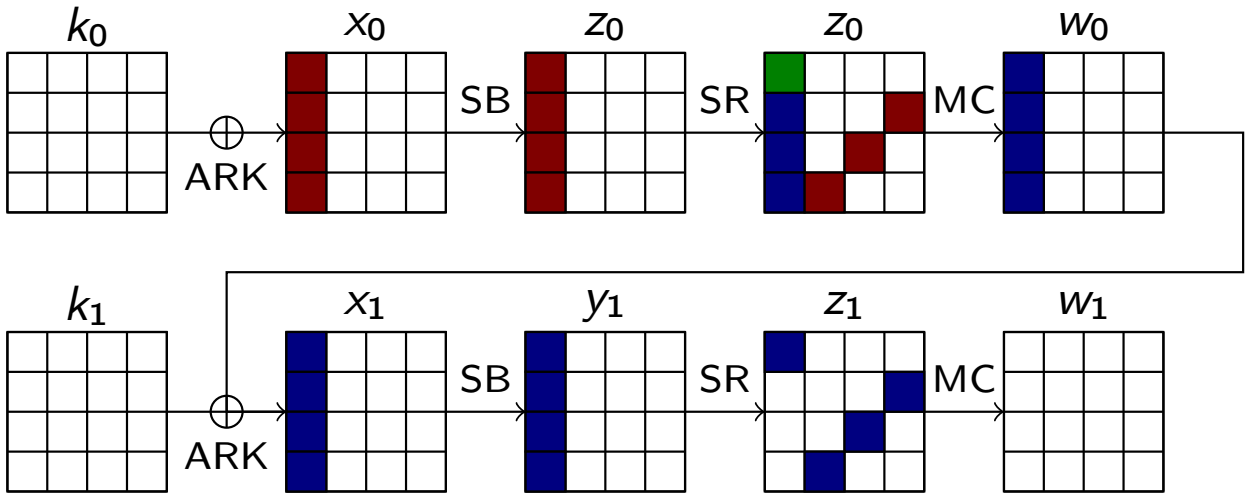
The case of AES

- **AES** is a block cipher with an 4×4 state of 8-bit S-Boxes (bytes).
- **The linear layer** permutes the bytes (Shiftrows) and mixes the columns (MixColumns).

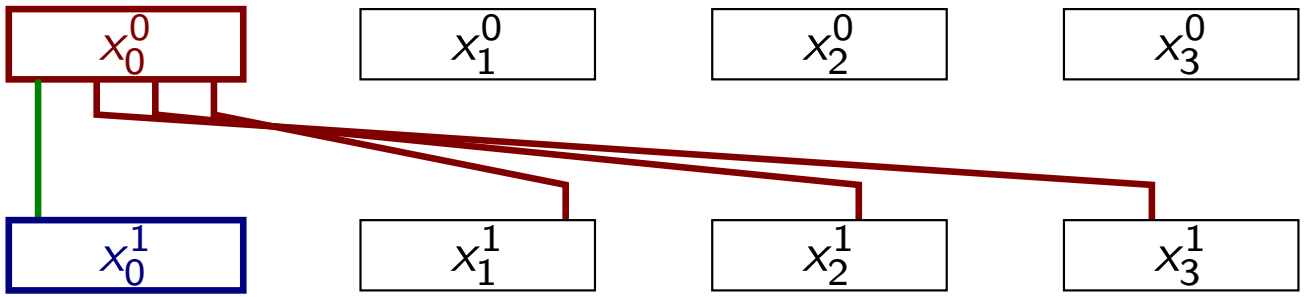


The Super S-Box

If we put together MC and SB, it creates a **Super S-Box** acting on $4 \times 8 = 32$ bits. This:



Looks like this:

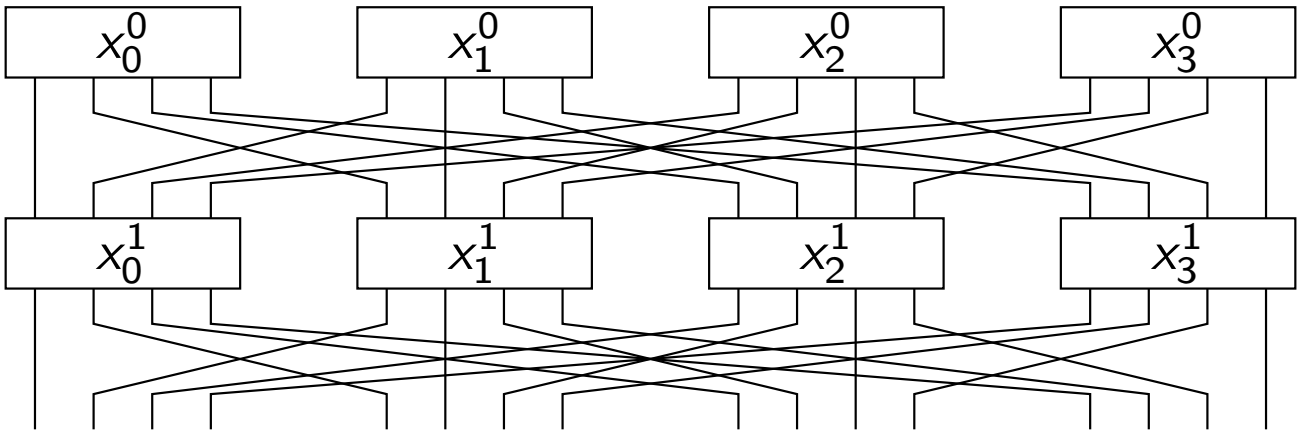


In our abstraction, **AES and Present look the same.**

Cell-based representation

- Remove any round constants
- Consider each S-Box as an arbitrary function S_j^i
- Replace each S-Box S_j^i by a **cell** x_j^i
- Each cell has a list of **possible assignments**: the table of S_j^i

- Only linear relations between cells remain
- We must find an **assignment** to **all cells** that satisfies **all linear relations**



Searching for a MITM attack

Solving the MITM problem

$\mathcal{R}[X]$:= all assignments of cells in X that satisfy the linear relations.

The MITM characteristic is defined by sets of cells:

- X_F: **forwards** ▶
- X_B: **backwards** ◀
- Merged: $X_M = X_F \cup X_B$ (covers a whole round)

The attack does:

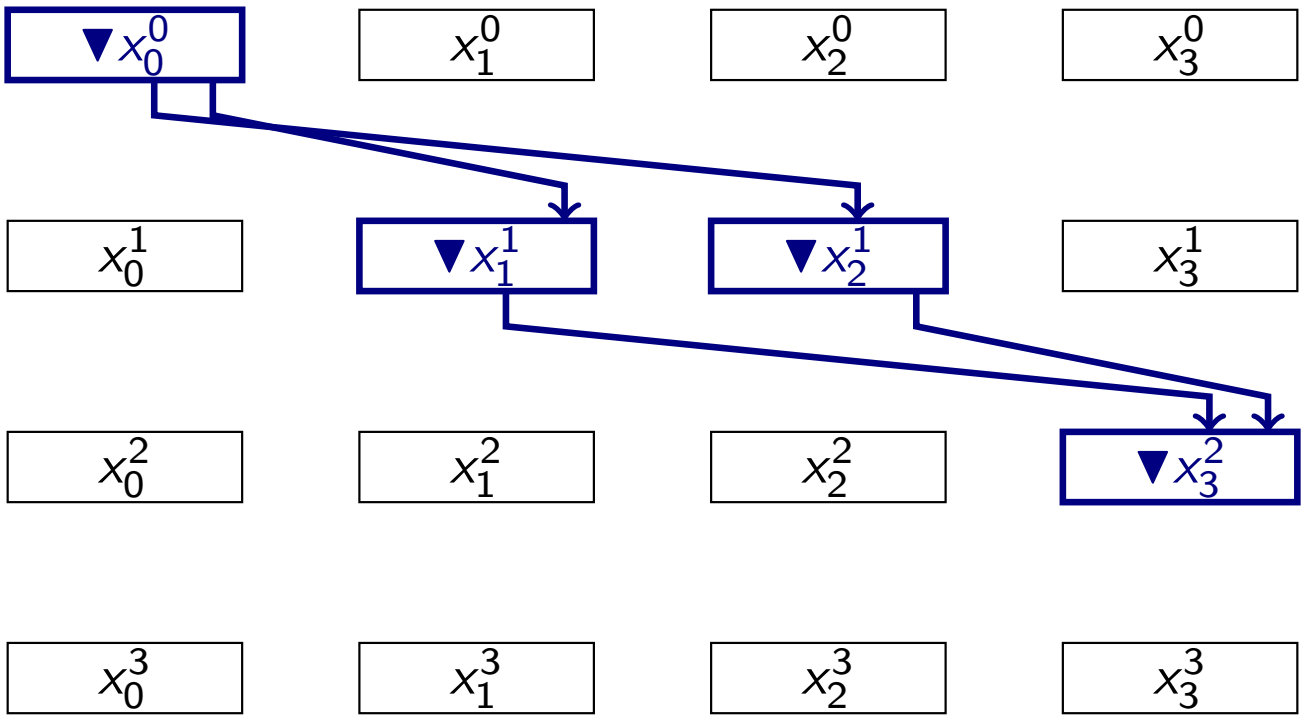
1. Compute $\mathcal{R}[X_F]$ (forward path)
2. Compute $\mathcal{R}[X_B]$ (backward path)
3. Compute $\mathcal{R}[X_M]$

The forward list $\mathcal{R}[X_F]$

We go **forwards**. Round by round, we guess the missing bits: 4 at rd. 0, 3 + 3 at rd. 1, 2 at rd. 2

- So, we have a valid assignment in time 1, memoryless.
- $\mathcal{R}[X_F]$ can be computed in $|\mathcal{R}[X_F]|$:

$$(\text{in } \log_2) \sum \text{weights of cells} - \sum \text{weights of edges} = 4 \times 4 - 4 = 12$$

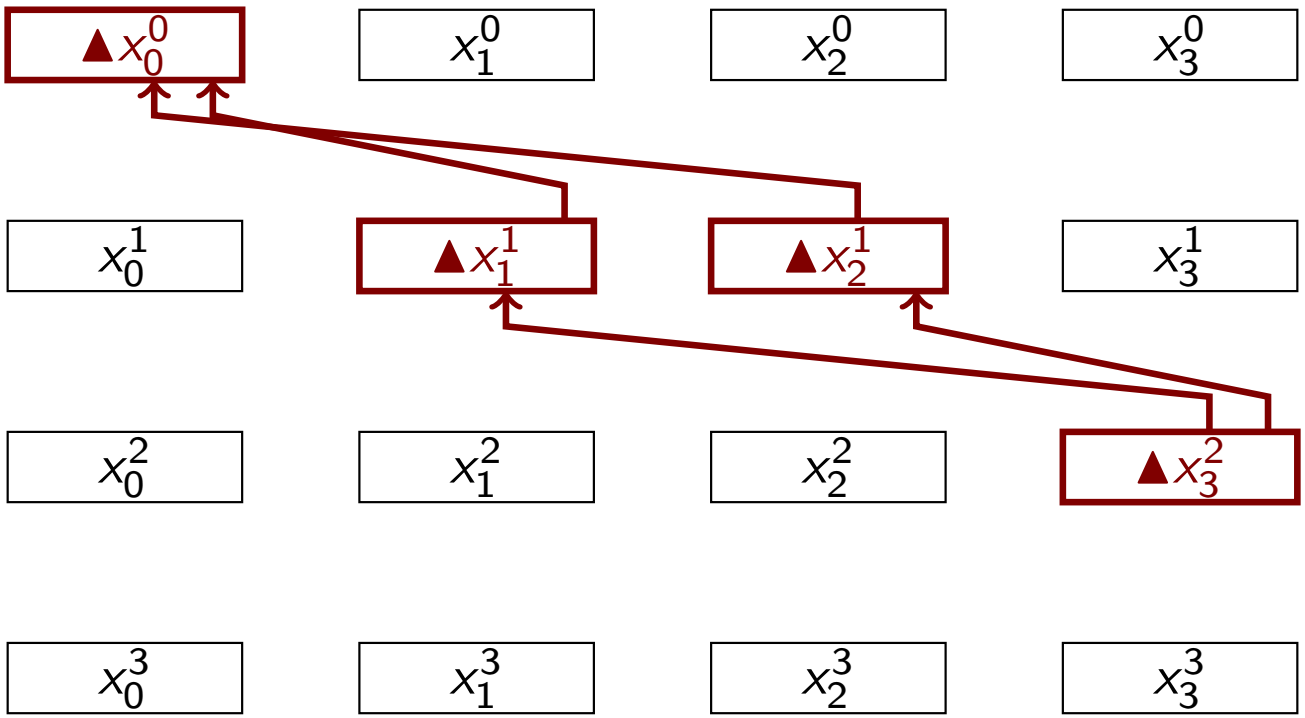


The backward list $\mathcal{R}[X_B]$

We go **backwards**. Round by round, we guess the missing bits: 4 at rd. 3, 3 + 3 at rd. 2, 2 at rd. 1

- So, we have a valid assignment in time 1, memoryless.
- $\mathcal{R}[X_B]$ can be computed in $|\mathcal{R}[X_B]|$:

$$(\text{in } \log_2) \sum \text{weights of cells} - \sum \text{weights of edges} = 4 \times 4 - 4 = 12$$

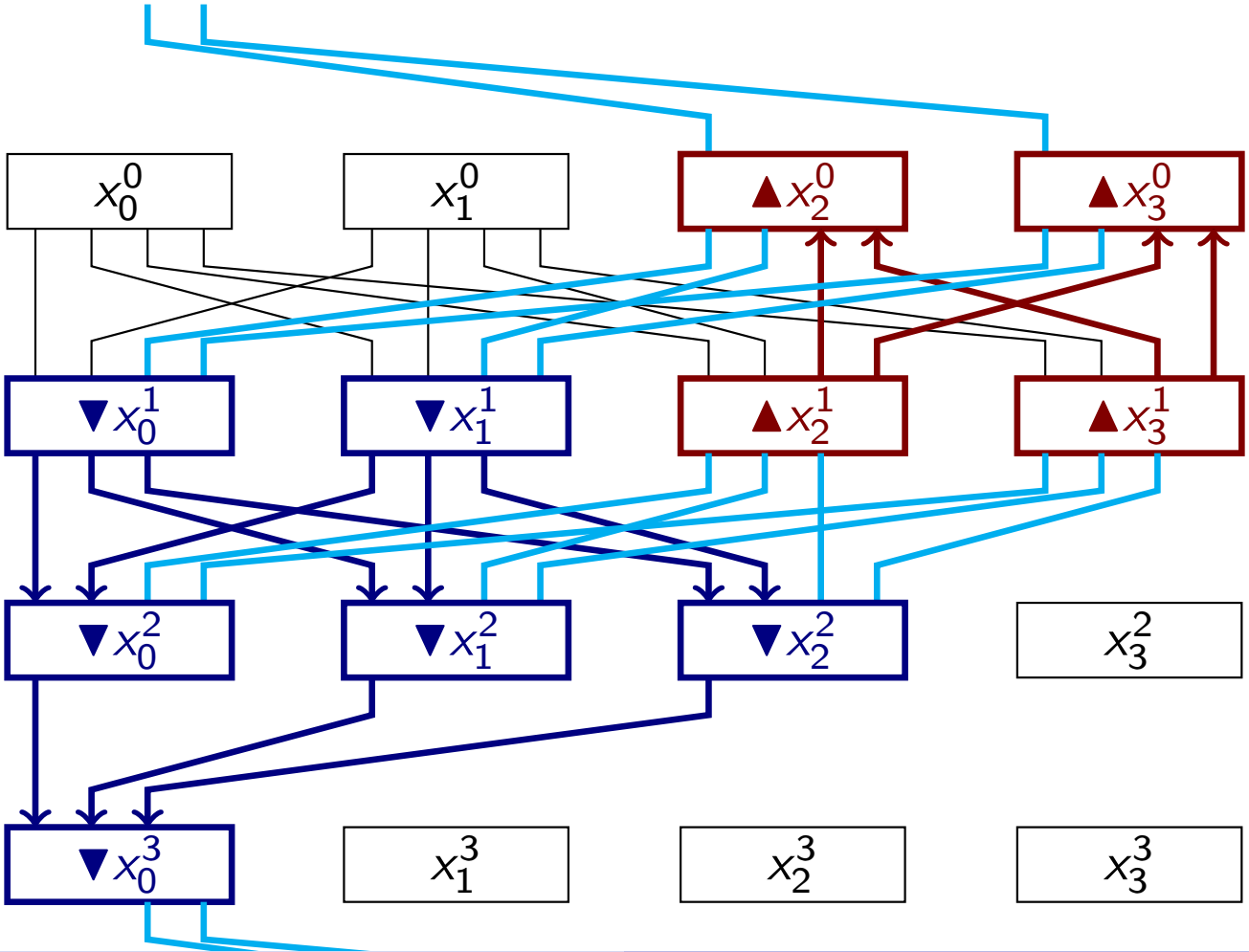


The merged list $\mathcal{R}[X_F \cup X_B]$

$$|\mathcal{R}[X_M]| = |\mathcal{R}[X_F]| \times |\mathcal{R}[X_B]|/(2^{\text{new edges}})$$

Forwards: 15 bits (3.75 cells); **Backwards:** 12 bits (3 cells)

Merged: $15 + 12 - 12 = 15$ bits (3.75 cells).



Attack complexity

- Classical time:

$$|\mathcal{R}[X_F]| + \max(|\mathcal{R}[X_B]|, |\mathcal{R}[X_M]|)$$

- Quantum time:

$$|\mathcal{R}[X_F]| + \sqrt{\max(|\mathcal{R}[X_B]|, |\mathcal{R}[X_M]|)}$$

- Classical memory:

$$|\mathcal{R}[X_F]|$$

- Quantum memory:

$$|\mathcal{R}[X_F]|$$

- the complexities depend on $|\mathcal{R}[X_F]|, |\mathcal{R}[X_B]|, |\mathcal{R}[X_M]|$
- the complexities depend only on X_F and X_B

MILP strategy

Search space: boolean variables for X_F and X_B

$\Downarrow$

Deduce the quantities $\ell_F := \log_2 |\mathcal{R}[X_F]|$, $\ell_B := \log_2 |\mathcal{R}[X_B]|$, $\ell_M := \log_2 |\mathcal{R}[X_M]|$ by linear inequalities

$\Downarrow$

Deduce the time and memory complexities (classical and quantum, in $\log_2$) of an attack based on X_F and X_B. **This is the objective function.**

Technical details

1. Reducing the memory

- Matching points of the form $\leftarrow\rightarrow$ can be turned into **global guesses** $\leftrightarrow$
- This precomputes some matches and reduces the list sizes

2. AES MixColumns

- In the AES case, the box is actually a (linear, MDS) MixColumns operation
- We can "match through MixColumns" to reduce $|\mathcal{R}[X_M]|$
- This can be modeled easily by making new cells appear in X_M

Attacking Haraka-512

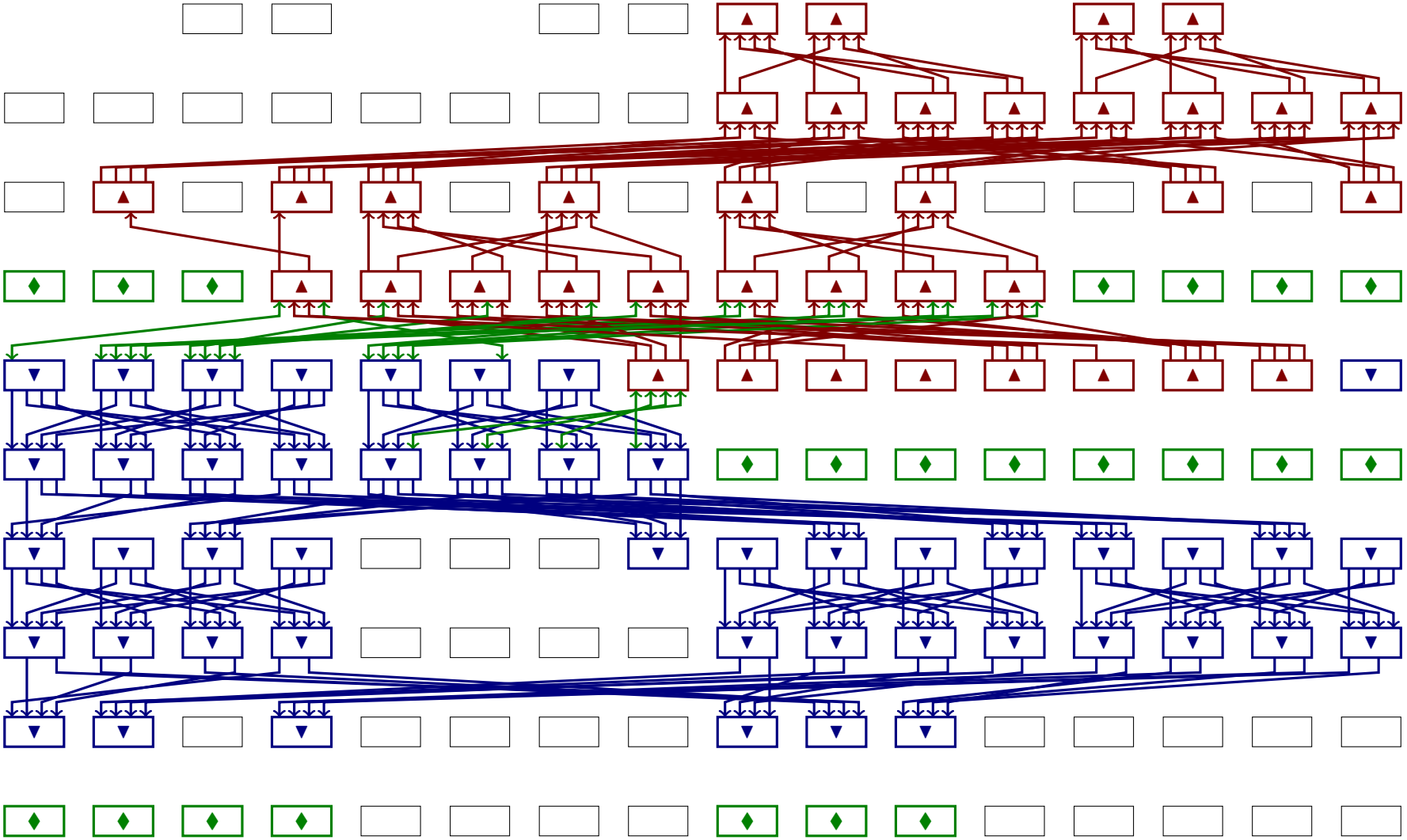
Attacking Haraka-512 v2

Ref.	Rounds	Model	Time	Memory
Bao et al. (EC 2021)	5.5/5	Classical	2^{240}	2^{128}
New	5.5/5	Classical	2^{240}	2^{16}
New	5.5/5	Quantum	$2^{123.34}$	2^{16}
New	**5/5**	**Classical**	2^{224}	2^{32}

- Our path matches at more rounds than **[BDGLSSW21]** and precomputes more matchings, which reduces the memory
- This path can now be used for a quantum attack

Attacking Haraka-512 v2 (ctd.)

Path for 5/5 rounds:



Attacking Haraka-512 v2 (ctd.)

A **low-memory** (full) MITM preimage attack can be a **partial preimage attack** that we **repeat many times.**

$\Longrightarrow$ for Haraka-512 v2, 64-bit partial preimages in about 2^{32} time and memory.

Input x (512 bits $= 4 \times 128$ bits)

e7 d4 9f 2d	16 f6 53 65	cd 99 33 01	d5 2a 66 a1
3f 05 c4 94	7a 61 37 17	6b 8c 47 1f	1f 10 08 cc
2e 53 f6 6a	5e 83 a9 6d	f0 7a b2 b9	69 a4 45 f4
b2 5c 0b 93	7a ee e2 c6	17 52 b3 74	e9 e4 79 f3

Output Haraka $-$ 512(x) (256 bits)

4d 35 de 97 63 ba c0 f0 4c dc 64 6b d1 e6 19 15

00 00 00 00 cb 51 f8 2b 9d 3e 50 e1 00 00 00 00

Conclusion

- Modeling MITM attacks can be **very simple** for permutations
- MITM attacks perform well in the quantum setting (up to a quadratic speedup)
- Extending our approach to the key-schedule path is an important question

Full version: ePrint 2022/189
Code: `github.com/AndreSchrottenloher/mitm-milp`

Thank you!