Simplified Modeling of MITM Attacks for Block Ciphers: new (Quantum) Attacks

André Schrottenloher[1] and Marc Stevens[2]

[1] Inria
[2] CWI

March 26, 2024

MITM problem

We consider a block cipher E with r rounds.

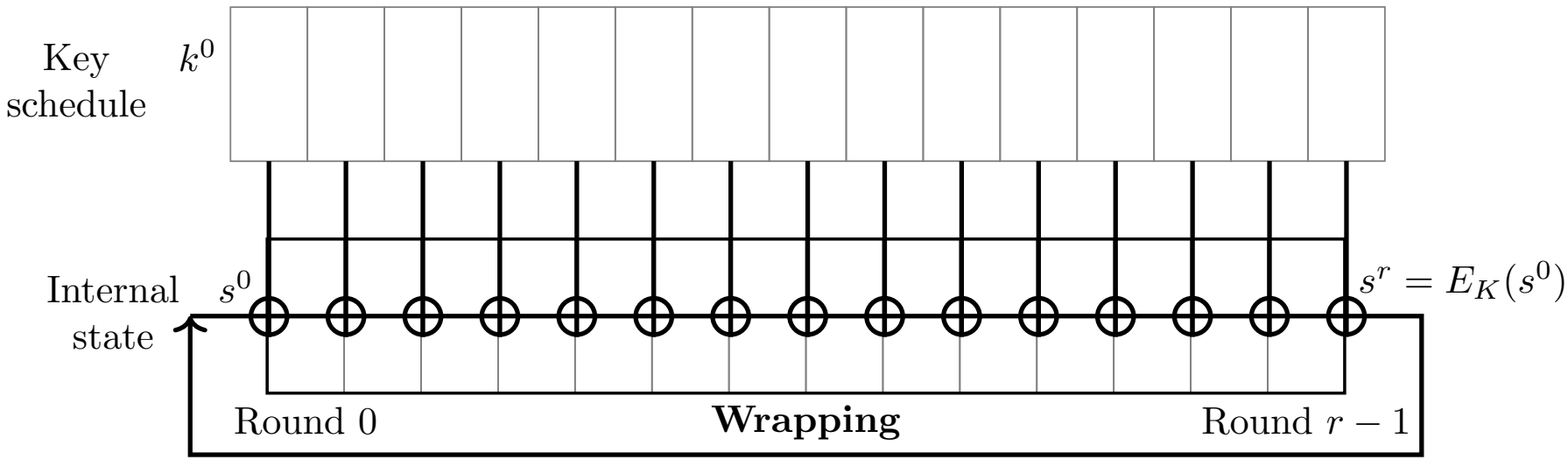
Find s^0 (state), K (key) satisfying a "wrapping" constraint.

- **Key-recovery attack:** we have access to the black-box E, constraint is:

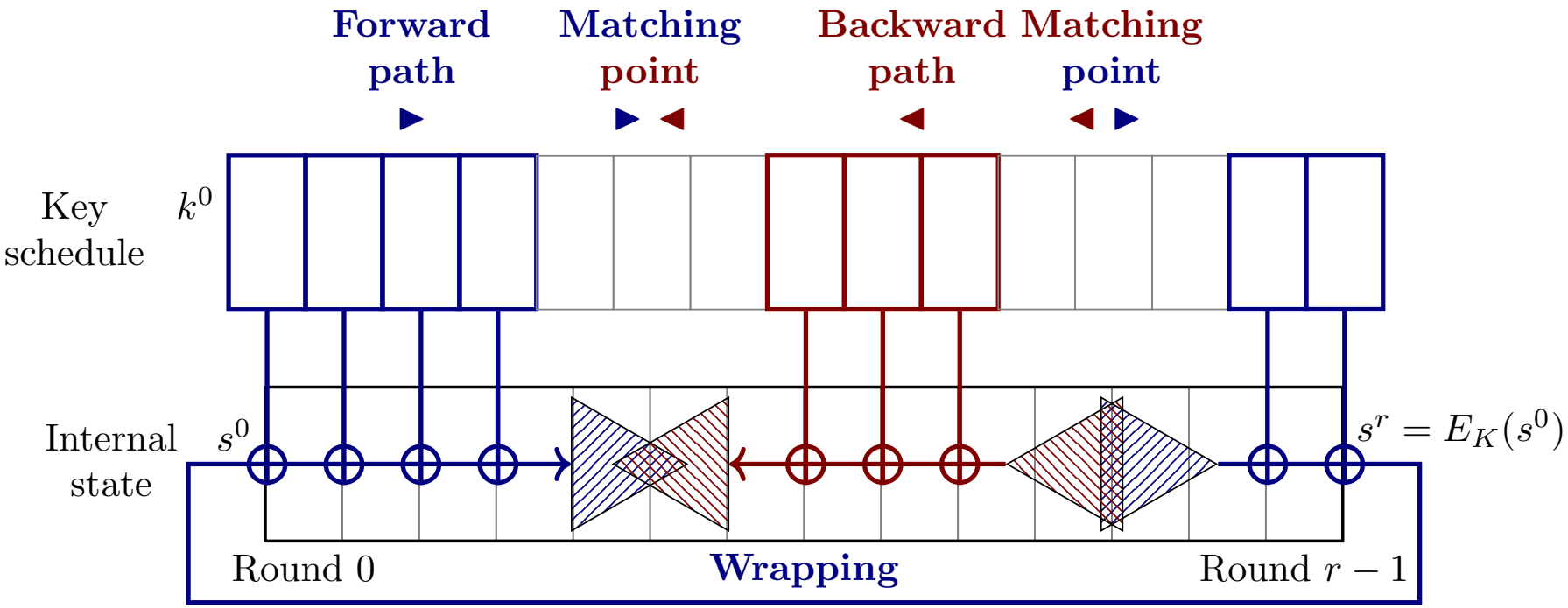
$$E_K(s^0) = E(s^0)$$

- **Pseudo-preimage attack:** just find K and s^0:

$$E_K(s^0) = s^0 \oplus T \ (T = \text{target preimage})$$



MITM attacks (ctd.)



1. Compute along a **forward computational path**
2. (Independently) compute along a **backward path**
3. Enumerate pairs of matching paths

Automatic search of MITM attacks

The attack is entirely defined by the choice of backward / forward paths

$\Longrightarrow$ define a set of choices

$\Longrightarrow$ optimize "attack complexity" within this set (using MILP)

- **[BGD+21]** and many others **[DHS+21,BGST22,QHD+23] ...**: define a complex set of rules that constrain the admissible paths
- **[SS22]**: simpler model, but only attacks permutations

This work: expands **[SS22]** with simple key-schedules (Saturnin, Present, etc.) and key-recovery attacks.

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

 S., Stevens. "Simplified MITM modeling for permutations: New (quantum) attacks.", CRYPTO 2022

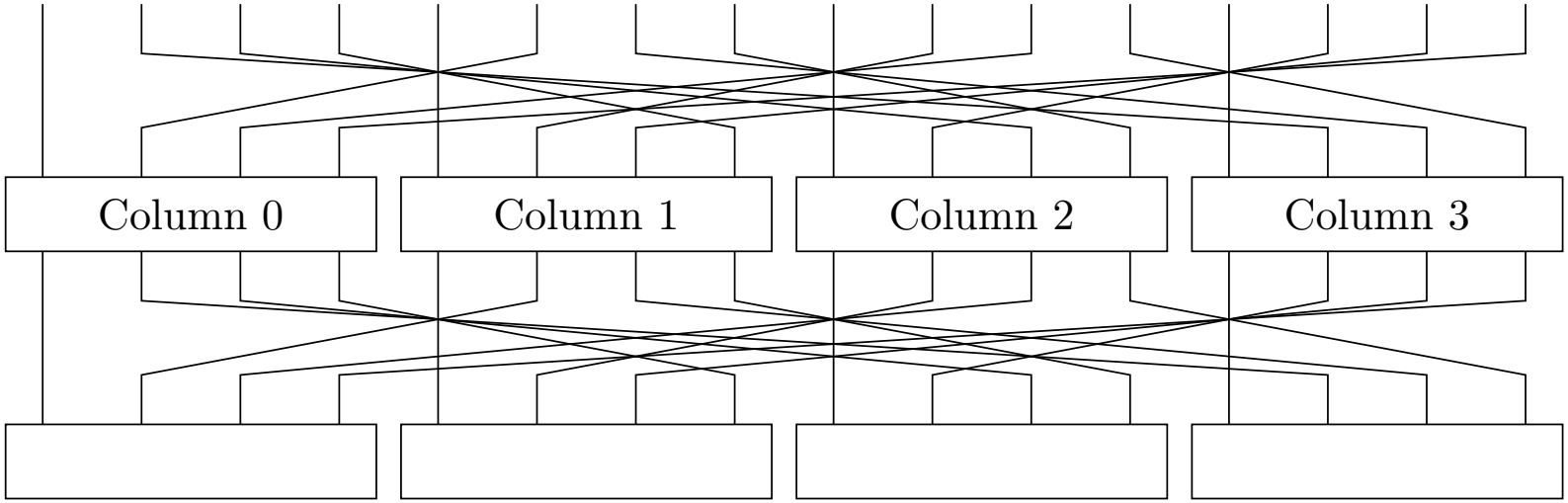
Outline

1. Keyless Model
2. Extending to Key-Schedules
3. Applications

Keyless Model

Abstracting the (SPN) cipher

- Basic operations: **S-Boxes** and **bit permutations**
- This includes AES-like ciphers thanks to the Super S-Box

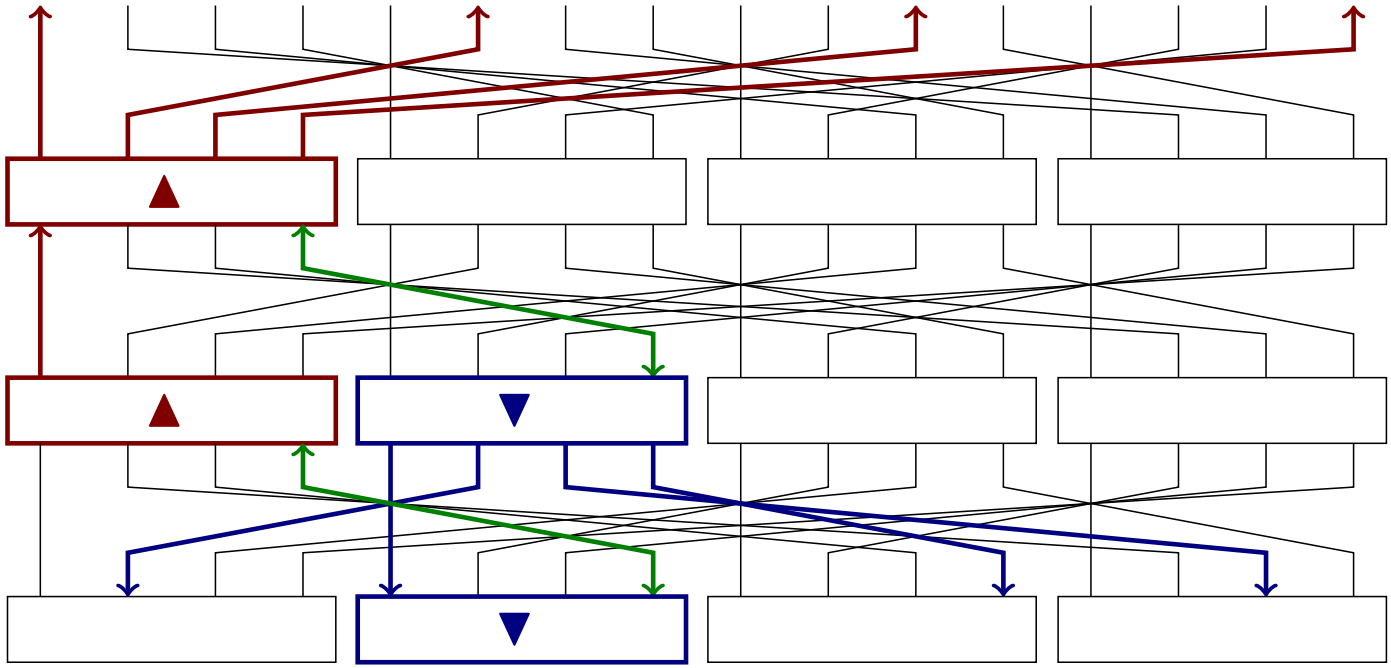


$\Longrightarrow$ S-Boxes are **nodes in an undirected graph**

$\Longrightarrow$ The "width" of an S-Box: how many bits / nibbles are necessary to compute it

A MITM characteristic

- Nodes are labeled **forward** or **backward** (or nothing)
- The **forward** / **backward list** contain all possibilities for the **forward** / **backward** paths
- The **merged list** is the list of pairs of paths, reduced using **matching points**



MITM attack complexity

- Classical time:

$$\ell_F + \max(\ell_B, \ell_M)$$

- Classical memory:

$$\ell_F$$

- Quantum time:

$$\ell_F + \sqrt{\max(\ell_B, \ell_M)}$$

- Quantum memory:

$$\ell_F$$

Computing the list sizes

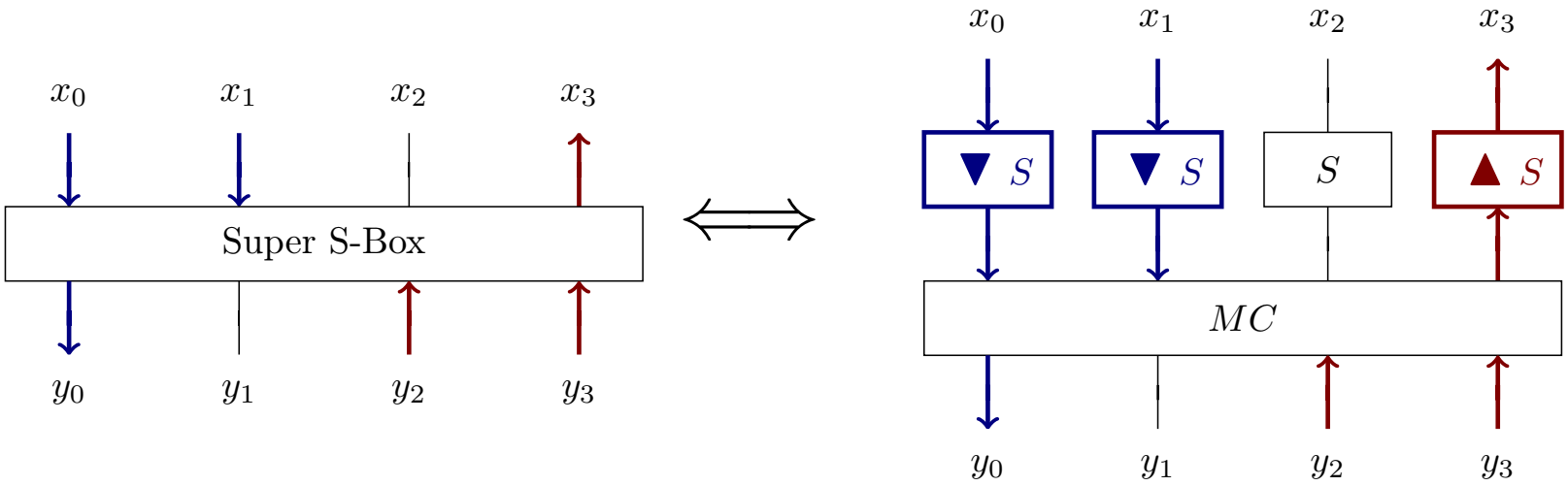
Number of choices for **forward** list ($\log_2$)
$=$ (total width of **forward** nodes) $-$ (number of edges between them)

Also works for **backward** & **merged** lists.

MILP modeling strategy:
Choice of nodes
$\Downarrow$
List sizes
$\Downarrow$
Complexity to minimize

Super S-Boxes

A **Super S-Box** is a node that behaves differently: we can match **through the node**.

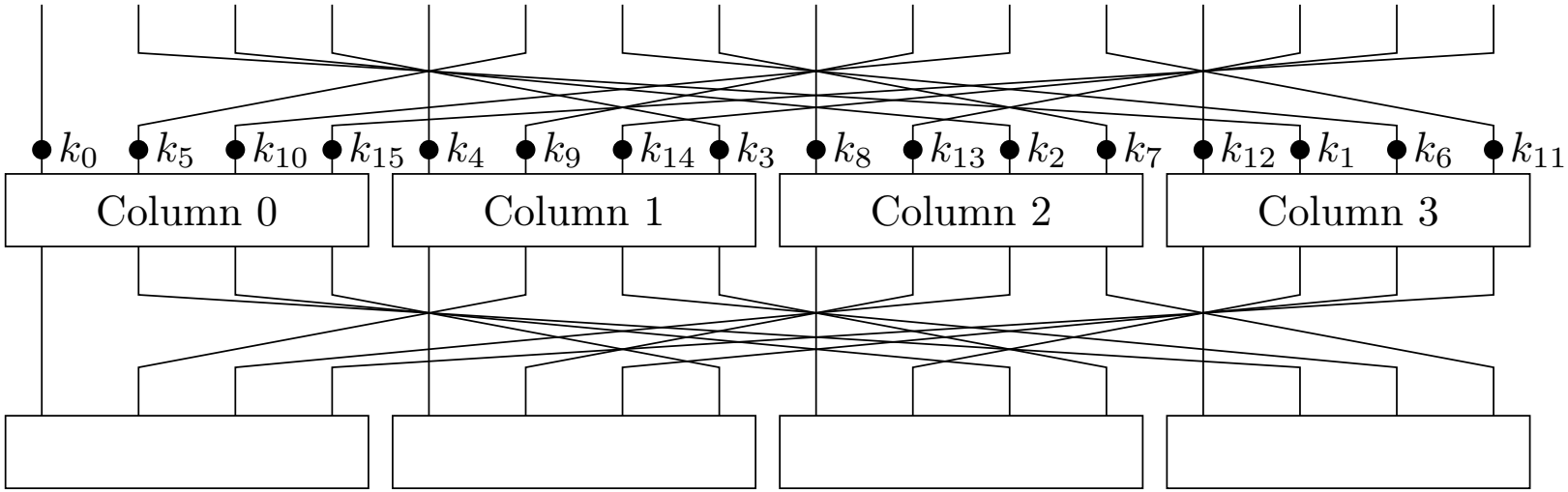


Ex. AES: If we know $c > 4$ edges in input and output, then we can match an amount of $c - 4$.

Extending to Key-Schedules

New variables & constraints

Key nibbles can now be XORed on any edge.



- We separate keys nibbles in: **forward**, **backward** and **shared** (known in both paths)
- They are counted in the respective lists

Modeling the key schedule

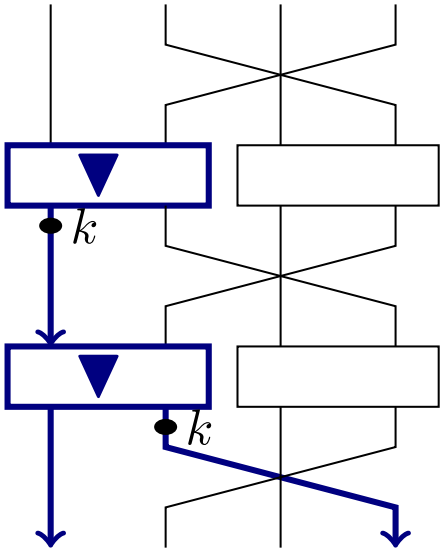
- Key schedule operations create new variables & relations (similar to the state path)
- We only support S-Boxes, permutations and selection of nibbles (e.g., Present)

Example:
$k_4|k_5|k_6|k_7 = S(k_0|k_1|k_2|k_3) \implies$ "if 4 key nibbles among $k_0, \ldots, k_7$ are **backward**, then all of them are **backward**".

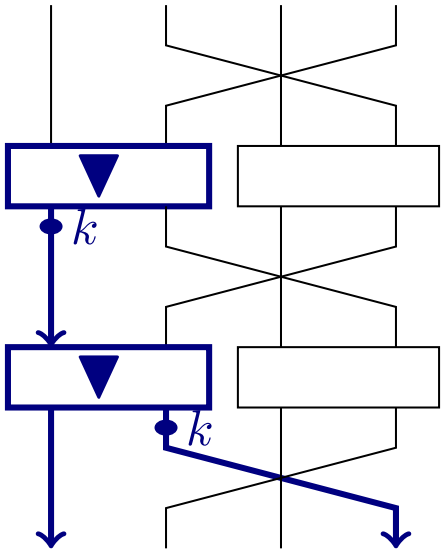
Modeling the key addition

If two nodes c, c' have the same color, then a key on the edge $c \to c'$ must have the same color.

Guessing one state nibble...



...cannot be better than guessing the key nibble instead



This basic constraint is adapted for Super S-Boxes.

Key-recovery case

Formulas for complexity & constraints slightly differ:

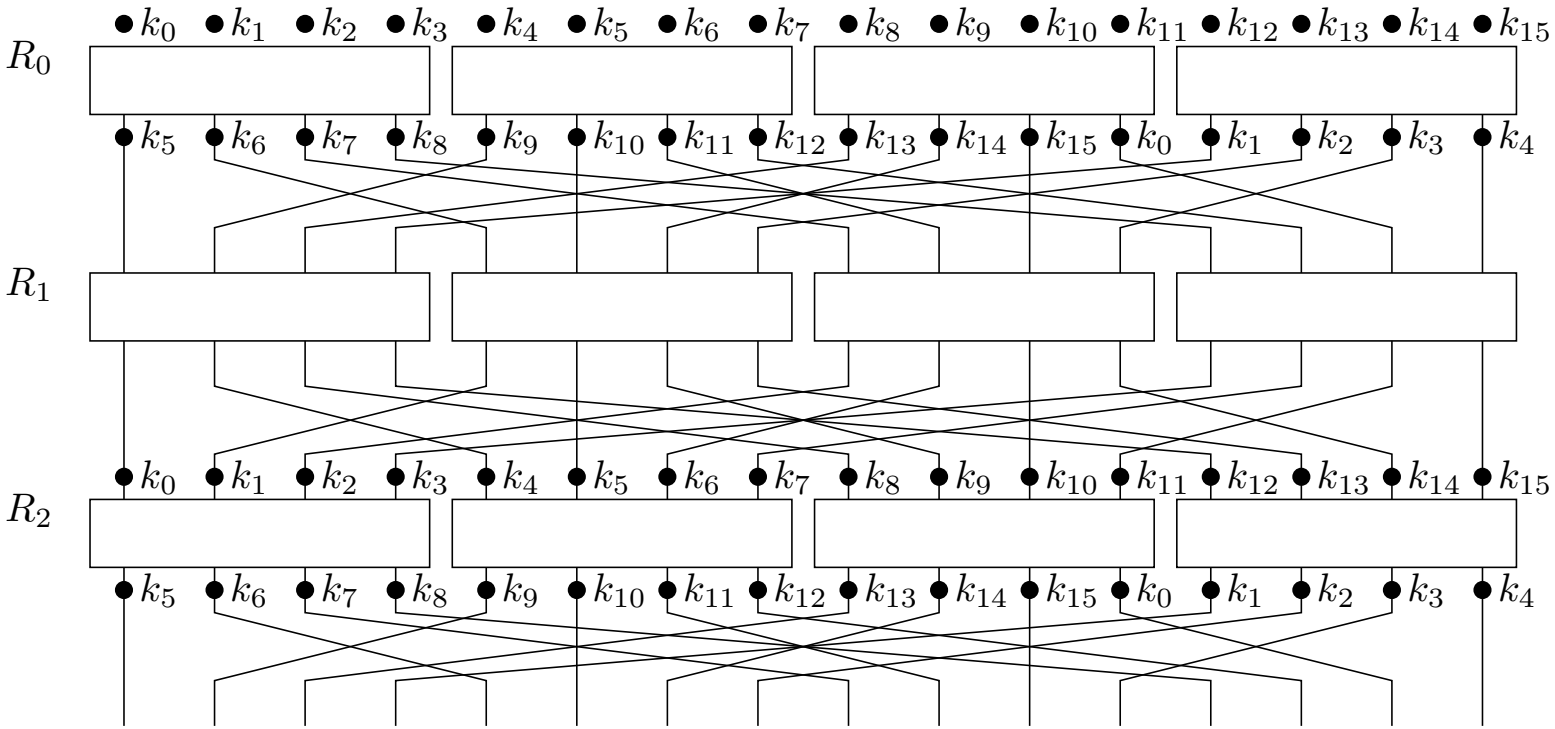
- In the preimage case, there are many solutions for K
- In the key-recovery case, we only have one solution for K: all must be explored

- The "wrapping" models the calls to the cipher by going through a big "cipher node"
- The data complexity can also be controlled

Applications

Example: Saturnin

- AES-like block cipher with 256-bit blocks and keys, 16-bit "super"nibbles
- Key-schedule: alternates K and rotated K



Canteaut, Duval, Leurent, Naya-Plasencia, Perrin, Pornin, S., "Saturnin: a suite of lightweight symmetric algorithms for post-quantum security." ToSC 2021

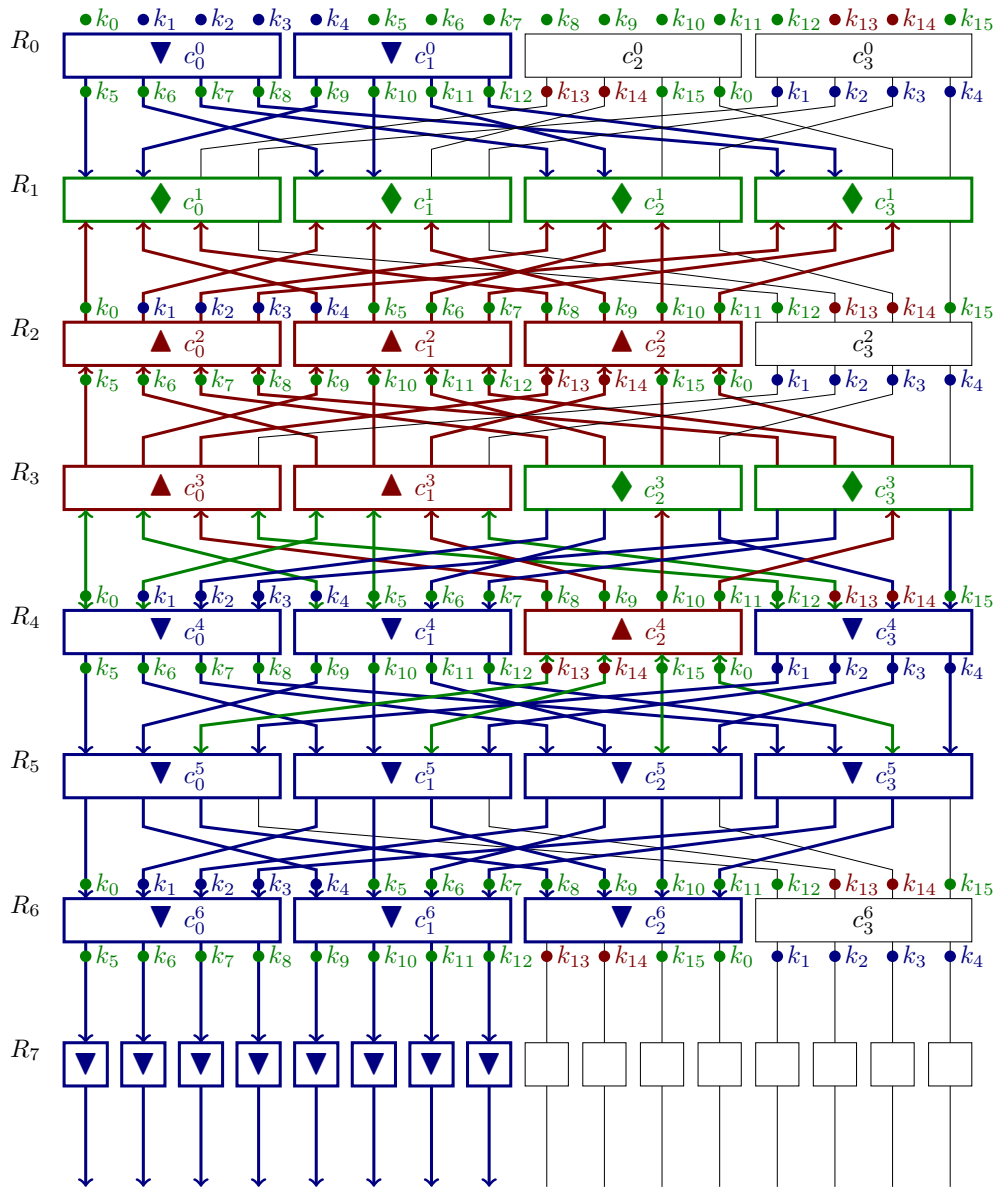
Results on pseudo-preimage

Given T, find s, K such that $E_{\mathsf{K}}(s) = s \oplus T$

	Rounds	Time	Memory	Reference
Classical	7 / 16	208 / 256	48	**[DHS+21]**
Classical	7 / 16	192 / 256	160	This work
Quantum	7 / 16	115.55 / 128	32 (QRAQM)	This work


Dong, Hua, Sun, Li, Wang, Hu. "Meet-in-the-middle attacks revisited: Key-recovery, collision, and preimage attacks", CRYPTO 2021

Example: quantum pseudo-preimage



Conclusion

- A simple MITM model for simple ciphers: very fast, when applicable
- New results on some lightweight designs (including Saturnin & quantum attacks)

Main open question:
Find the "best way" to handle key-schedules like AES, which create complex linear relations in the paths.

Paper: `doi.org/10.46586/tosc.v2023.i3.146-183`
Code: `github.com/AndreSchrottenloher/key-mitm`

Thank you!