ShiftRows Alternatives for AES-like Ciphers and Optimal Cell Permutations for Midori and Skinny

Gianira N. Alfarano[1], Christof Beierle[2], Takanori Isobe[3], **Stefan Kölbl**[4], Gregor Leander[2]

March 25th, 2019

[1]University of Zurich, Switzerland

[2]Horst Görtz Institute for IT Security, Ruhr-Universität Bochum, Germany

[3]University of Hyogo, Japan

[4]Cybercrypt, Denmark

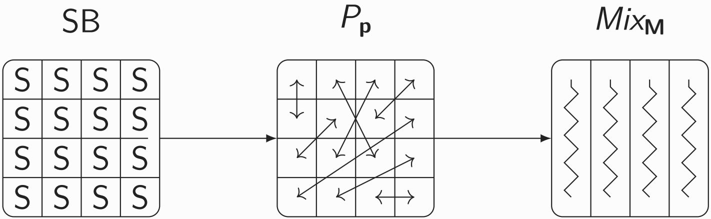
AES-like Primitives

AES-like Constructions are *very* popular

- Block Ciphers:
  - Deoxys-BC, Kuzneychik, LED, Midori, Prince, Skinny, ...
- Hash Functions:
  - Grøstl, Photon, Streebog, Whirlpool, ...
- Permutations:
  - AESQ, Haraka, Prøst, Simpira, ...

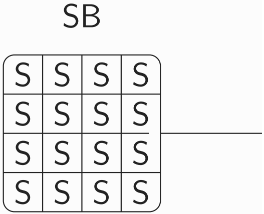
AES-like Primitives

Building blocks:



AES-like Primitives

Building blocks:

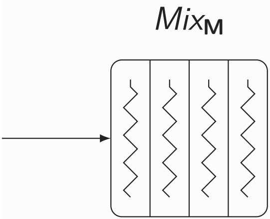


- Apply S-box on each cell
- Only non-linear component
- Vast area of research

AES-like Primitives

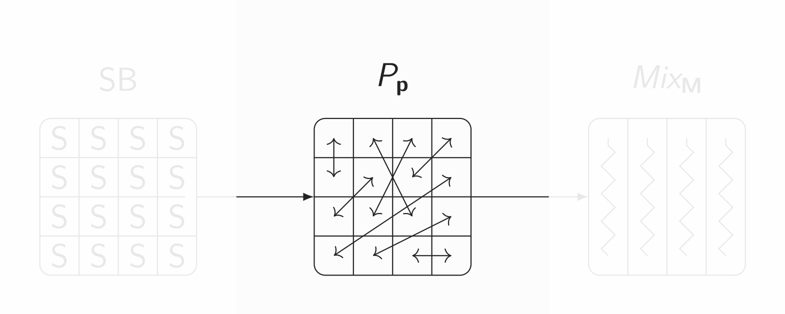
Building blocks:

- Multiply each column with matrix
- Vast area of research



AES-like Primitives

Building blocks:



Security of AES-like Primitives

Resistance against differential and linear cryptanalysis.

- S-box: Every *active* S-box has an effect on probability of differential trail.
- Mix: Gives a lower bound on active S-boxes in one round.
- Permute: Heavily influences bounds for multiple rounds.

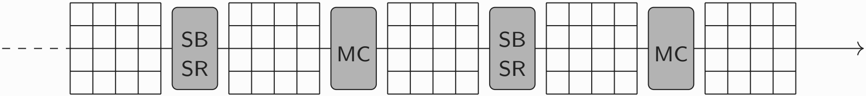
Goal

Find a lower bound on the number of active S-boxes for a design.

Security of AES-like Primitives

Example AES

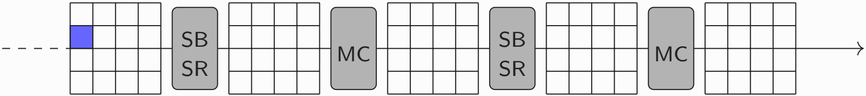
- MixColumns has *branch number* 5.
- Only constraint active input $+$ output ≥ 5.



Security of AES-like Primitives

Example AES

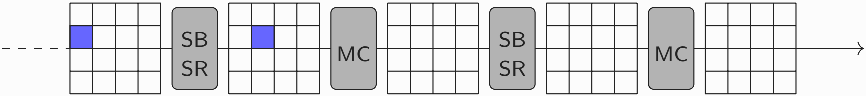
- MixColumns has *branch number* 5.
- Only constraint active input $+$ output ≥ 5.



Security of AES-like Primitives

Example AES

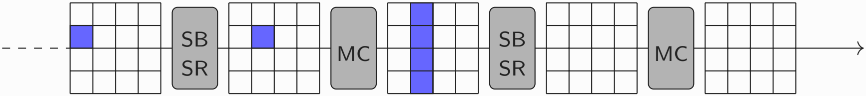
- MixColumns has *branch number* 5.
- Only constraint active input + output ≥ 5.



Security of AES-like Primitives

Example AES

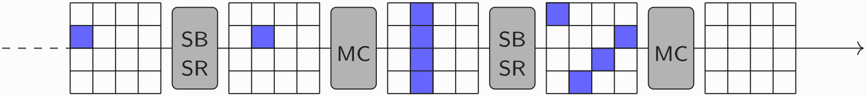
- MixColumns has *branch number* 5.
- Only constraint active input $+$ output ≥ 5.



Security of AES-like Primitives

Example AES

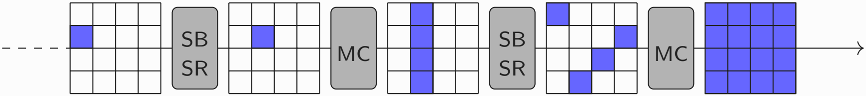
- MixColumns has *branch number* 5.
- Only constraint active input + output ≥ 5.



Security of AES-like Primitives

Example AES

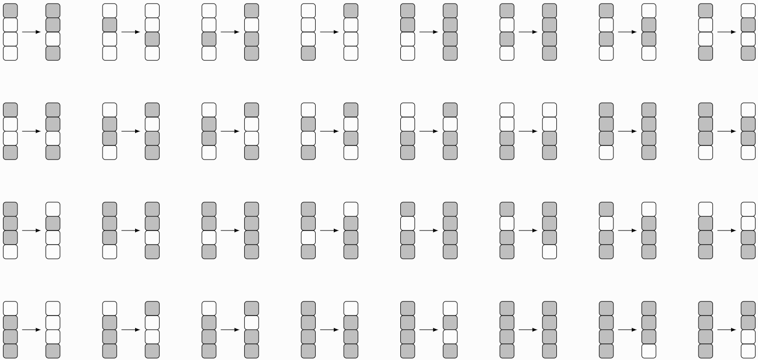
- MixColumns has *branch number* 5.
- Only constraint active input + output ≥ 5.



Security of AES-like Primitives

Can be much more complex for other choices:

- Midori (Branch number 4)
- but not possible to have $2 \to 3$ (or $3 \to 2$) transitions.
- Skinny (Branch number 2)



AES-like Primitives

Known results on the permute layer

- M is MDS and $n \times n$ state $\rightarrow$ AES ShiftRows optimal
- *Linear Frameworks for Block Ciphers*, Daemen, Knudsen, Rijmen, DCC, 2001
- *Analyzing Permutations for AES-like Ciphers: Understanding ShiftRows*, Beierle, Jovanovic, Lauridsen, Leander, Rechberger, CT-RSA, 2015

AES-like Primitives

Known results on the permute layer

- M is MDS and $n \times n$ state $\rightarrow$ AES ShiftRows optimal
- *Linear Frameworks for Block Ciphers*, Daemen, Knudsen, Rijmen, DCC, 2001
- *Analyzing Permutations for AES-like Ciphers: Understanding ShiftRows*, Beierle, Jovanovic, Lauridsen, Leander, Rechberger, CT-RSA, 2015

Problem we solve

Given an $n \times m$ state of w-bit words with a fixed SB and Mix layer. What is the optimal choice for permute w.r.t. security against differential/linear cryptanalysis?

Security of AES-like Primitives

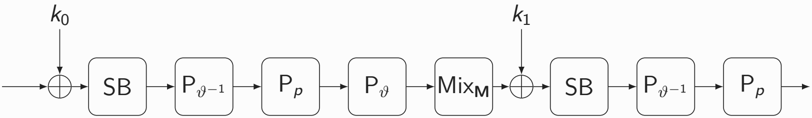
How can we find the optimal choice for p?

- For a 4×4 state we already get $2^{44.25}$ choices.
- Need to evaluate cryptanalytical properties for all of them?
- How can we limit the search space?

Classifying Cell Permutations

First observation:

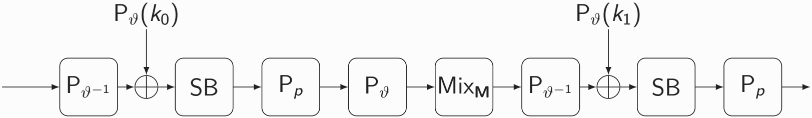
- Consider permutation p and ϑ.
- If $\mathsf{Mix}_{\mathbf{M}} \circ \mathsf{Permute}_{\vartheta} = \mathsf{Permute}_{\vartheta} \circ \mathsf{Mix}_{\mathbf{M}}$...
- ...then $\mathsf{Permute}_{\mathsf{p}}$ and $\mathsf{Permute}_{\vartheta \circ \mathsf{p} \circ \vartheta^{-1}}$ have the same cryptographic properties.



Classifying Cell Permutations

First observation:

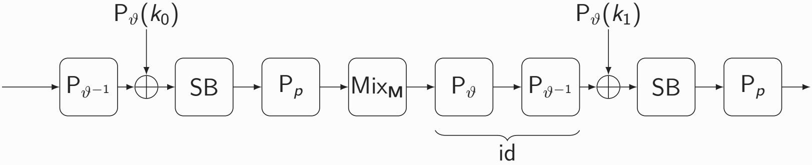
- Consider permutation p and ϑ.
- If $\mathsf{Mix}_{\mathbf{M}} \circ \mathsf{Permute}_{\vartheta} = \mathsf{Permute}_{\vartheta} \circ \mathsf{Mix}_{\mathbf{M}}$...
- ...then $\mathsf{Permute}_{\mathbf{p}}$ and $\mathsf{Permute}_{\vartheta\circ\mathbf{p}\circ\vartheta^{-1}}$ have the same cryptographic properties.



Classifying Cell Permutations

First observation:

- Consider permutation p and ϑ.
- If $\mathrm{Mix}_{\mathbf{M}} \circ \mathrm{Permute}_{\vartheta} = \mathrm{Permute}_{\vartheta} \circ \mathrm{Mix}_{\mathbf{M}}$...
- ...then $\mathrm{Permute}_{\mathsf{p}}$ and $\mathrm{Permute}_{\vartheta \circ \mathsf{p} \circ \vartheta^{-1}}$ have the same cryptographic properties.



Classifying Cell Permutations

Equivalence Relation:

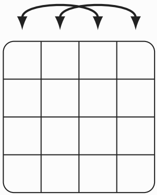
- Two permutations p, p' are $\mathbf{M}$-equivalent if there exists ϑ such that
$$p' = \vartheta \circ p \circ \vartheta^{-1}, \tag{1}$$
and ϑ commutes with $\mathbf{M}$.
- **M**-equivalent permutations will have same number of active S-boxes!
- Unclear how to efficiently determine **M**-equivalence.

Classifying Cell Permutations

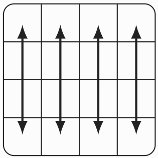
weak M-equivalence:

- $\vartheta = \pi \circ \phi$
- π permutes whole columns of the state
- ϕ permutes insides columns individually

π



ϕ



Classifying Cell Permutations

Structure matrix

Example

$$\begin{bmatrix} 0 & 4 & 8 & 12 \\ 1 & 5 & 9 & 13 \\ 2 & 6 & 10 & 14 \\ 3 & 7 & 11 & 15 \end{bmatrix} \overset{p}{\mapsto} \begin{bmatrix} 4 & 0 & 13 & 1 \\ 5 & 6 & 14 & 2 \\ 11 & 9 & 8 & 3 \\ 15 & 12 & 7 & 10 \end{bmatrix}, \mathbf{A}_p = \begin{pmatrix} 0 & 1 & 0 & 3 \\ 2 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 2 & 0 \end{pmatrix}$$

Classifying Cell Permutations

Structure matrix

Example

$$\begin{bmatrix} 0 & 4 & 8 & 12 \\ 1 & 5 & 9 & 13 \\ 2 & 6 & 10 & 14 \\ 3 & 7 & 11 & 15 \end{bmatrix} \overset{p}{\mapsto} \begin{bmatrix} 4 & 0 & 13 & 1 \\ 5 & 6 & 14 & 2 \\ 11 & 9 & 8 & 3 \\ 15 & 12 & 7 & 10 \end{bmatrix}, \mathbf{A}_p = \begin{pmatrix} 0 & 1 & 0 & 3 \\ 2 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 2 & 0 \end{pmatrix}$$

Classifying Cell Permutations

Structure matrix

Example

$$\begin{bmatrix} 0 & 4 & 8 & 12 \\ 1 & 5 & 9 & 13 \\ 2 & 6 & 10 & 14 \\ 3 & 7 & 11 & 15 \end{bmatrix} \overset{p}{\mapsto} \begin{bmatrix} 4 & 0 & 13 & 1 \\ 5 & 6 & 14 & 2 \\ 11 & 9 & 8 & 3 \\ 15 & 12 & 7 & 10 \end{bmatrix}, \mathbf{A}_p = \begin{pmatrix} 0 & 1 & 0 & 3 \\ 2 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 2 & 0 \end{pmatrix}$$

Classifying Cell Permutations

Structure matrix

Example

$$\begin{bmatrix} 0 & 4 & 8 & 12 \\ 1 & 5 & 9 & 13 \\ 2 & 6 & 10 & 14 \\ 3 & 7 & 11 & 15 \end{bmatrix} \overset{p}{\mapsto} \begin{bmatrix} 4 & 0 & 13 & 1 \\ 5 & 6 & 14 & 2 \\ 11 & 9 & 8 & 3 \\ 15 & 12 & 7 & 10 \end{bmatrix}, \mathbf{A}_p = \begin{pmatrix} 0 & 1 & 0 & 3 \\ 2 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 2 & 0 \end{pmatrix}$$

Classifying Cell Permutations

Structure matrix

Example

$$\begin{bmatrix} 0 & 4 & 8 & 12 \\ 1 & 5 & 9 & 13 \\ 2 & 6 & 10 & 14 \\ 3 & 7 & 11 & 15 \end{bmatrix} \overset{p}{\mapsto} \begin{bmatrix} 4 & 0 & 13 & 1 \\ 5 & 6 & 14 & 2 \\ 11 & 9 & 8 & 3 \\ 15 & 12 & 7 & 10 \end{bmatrix}, \mathbf{A}_p = \begin{pmatrix} 0 & 1 & 0 & 3 \\ 2 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 2 & 0 \end{pmatrix}$$

Classifying Cell Permutations

Result

We provide an efficient algorithm to enumerate all permutations up to weak $\mathbf{M}$-equivalence.

Basic idea of the algorithm:

- Enumerates all permutations up to **weak M** equivalence for given structure matrix.
- For example 4×4 state there are 10147 valid structure matrices.
- Find *smallest* representatives of each equivalence class.

Classifying Cell Permutations

When does **weak M** imply **M** equivalence?

- Consider the matrix $\mathbf{M}$.
- Let G be the directed graph corresponding to the adjacency matrix of M.
- If G is strongly connected then $\mathbf{M}$ coincides with weak $\mathbf{M}$.

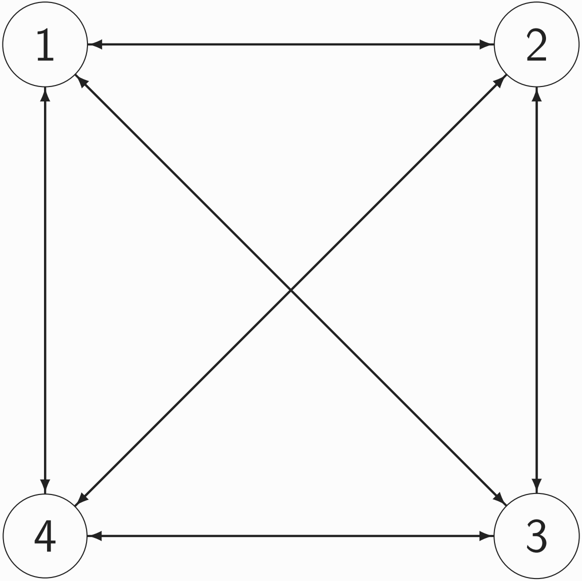
Case Study: Midori

Midori block cipher

- Energy efficient cipher
- 4×4 state
- Uses generic *p*
- MixColumns (Branch number 4, not all transitions possible)

$$\mathbf{M} = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}.$$

Case Study: Midori



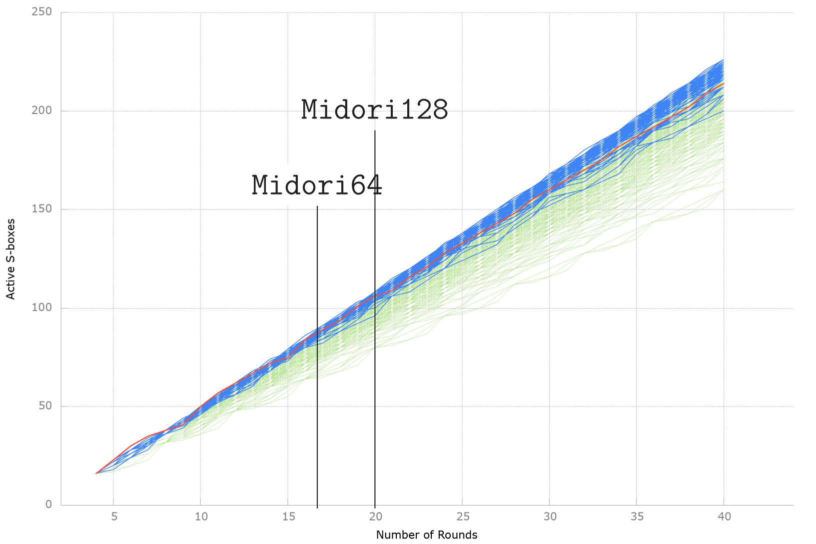
Case Study: Midori

Takes a few days on a standard PC to find all permutations up to **M**-equivalence.

- $2^{21.7}$ distinct equivalence classes.
- MILP (slow for larger number of rounds)
- Using branch and bound (Matsui's algorithm) much faster

`https://github.com/kste/matsui`

Case Study: Midori



Case Study: Midori

Conclusion

- Original permutation optimal for 1 to 12 rounds
- ...except for 9 rounds: 44 active S-boxes (instead of 41).
- For any higher number of rounds it is never optimal.

Case Study: Midori

Proof in the paper

- If p, p^2 and p^3 have the structure matrix

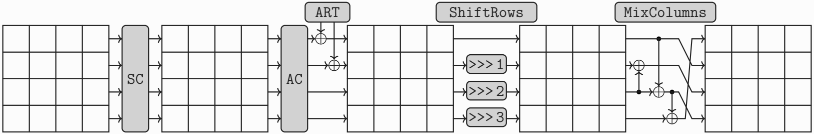
$$\mathbf{A}_p = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix} \tag{2}$$

  then there are at least 28 active S-boxes for 6 rounds.

Case Study: Skinny

Skinny

- Lightweight Tweakable Block Cipher
- Uses AES ShiftRows
- MixColumns (Branch number 2)



Case Study: Skinny

Results using our algorithm

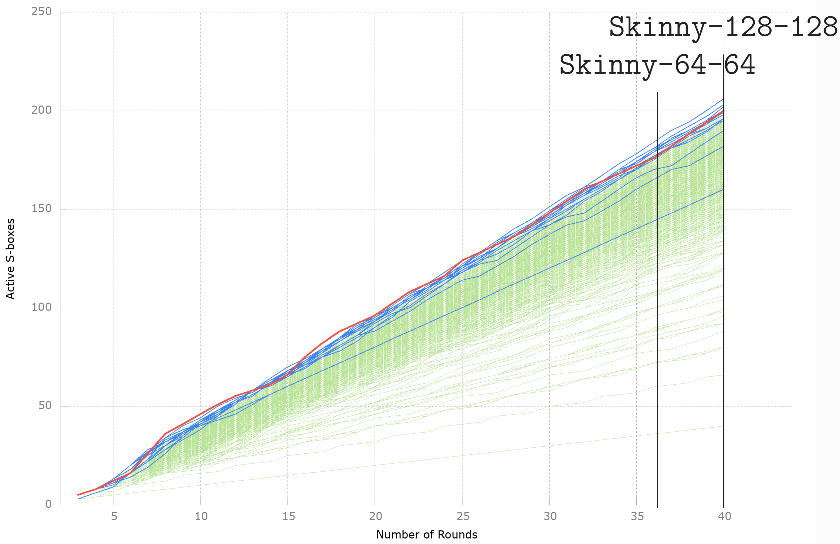
- weak M also implies M for Skinny MixColumns
- In total $2^{39.66}$ equivalence classes.
- Took 23.8 CPU days to find them.

Case Study: Skinny

We filter further:

- Only use permutations which give good diffusion
- Still 2.726.526 left...
- $\approx$ 2937 CPU days to run Matsui's for all variants

Case Study: Skinny



Conclusion

Summary

- Better theoretical understanding
- Useful tool for future designs
- Possible to evaluate the *best* choice for some designs