Rebound Attack on Reduced-Round Versions of JH

Vincent Rijmen Deniz Toz Kerem Varıcı

K.U.Leuven, Dept. of Electrical Engineering, ESAT/SCD/COSIC and IBBT

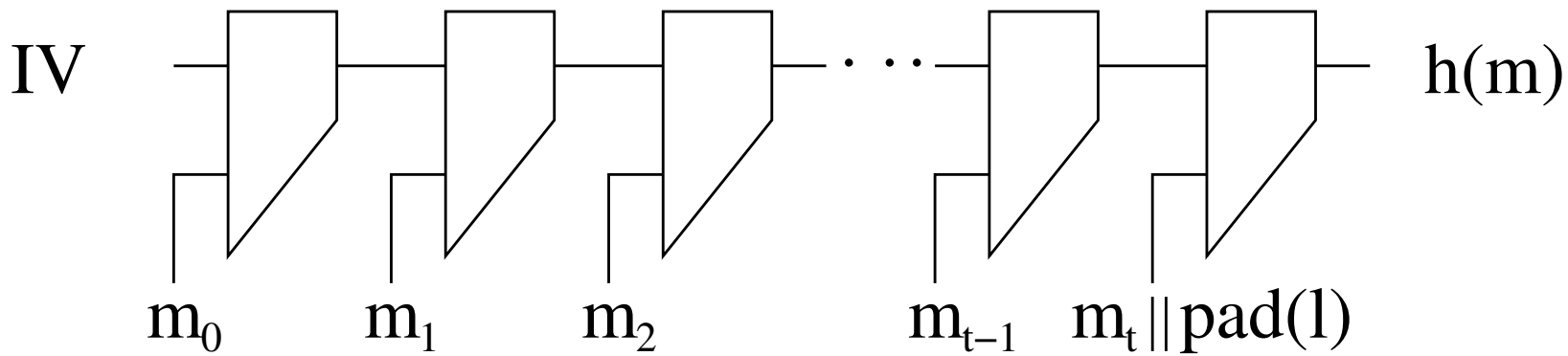
February 9, 2010

Outline

Outline

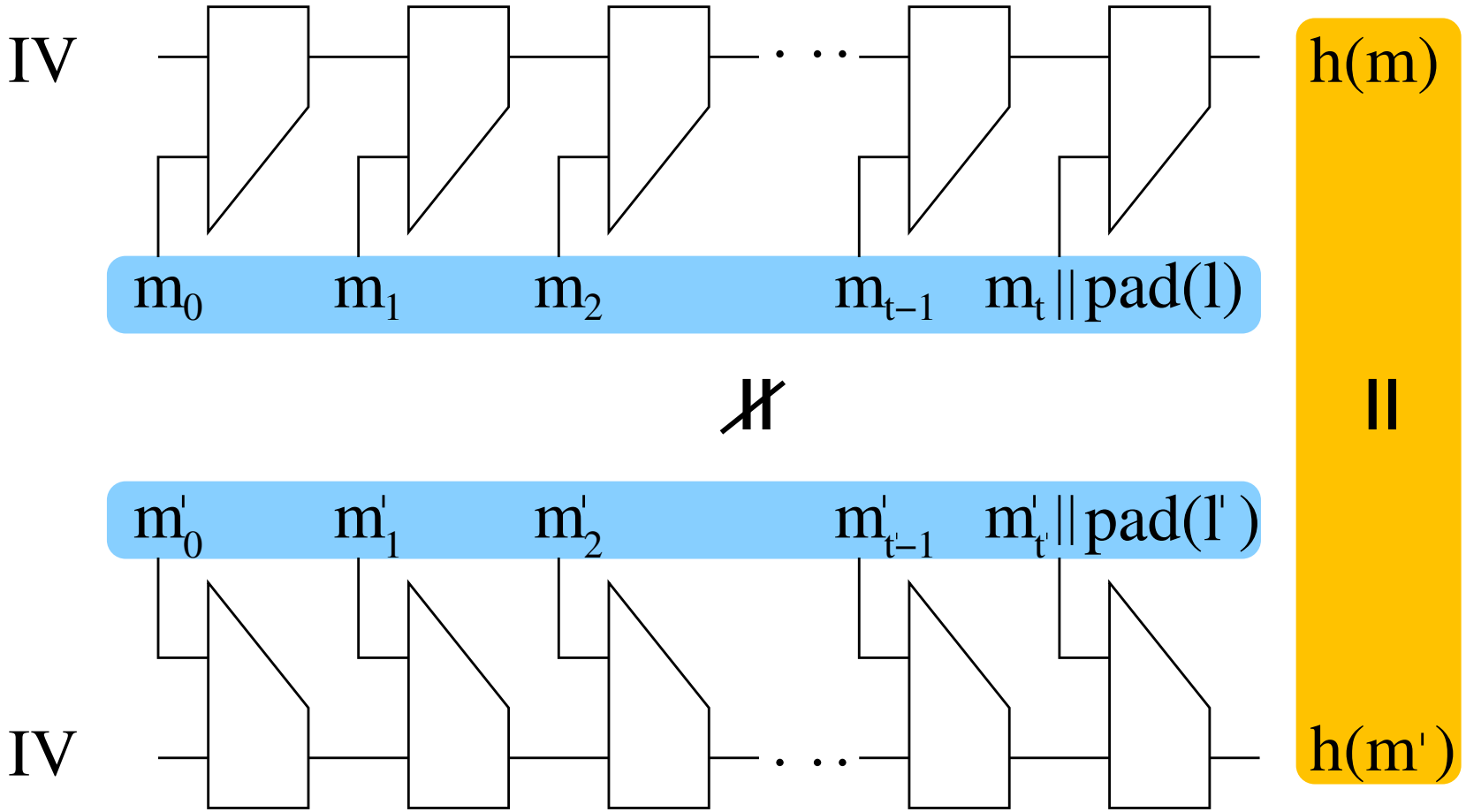
Collision Attacks on Iterated Hash Functions

Iterated hash function



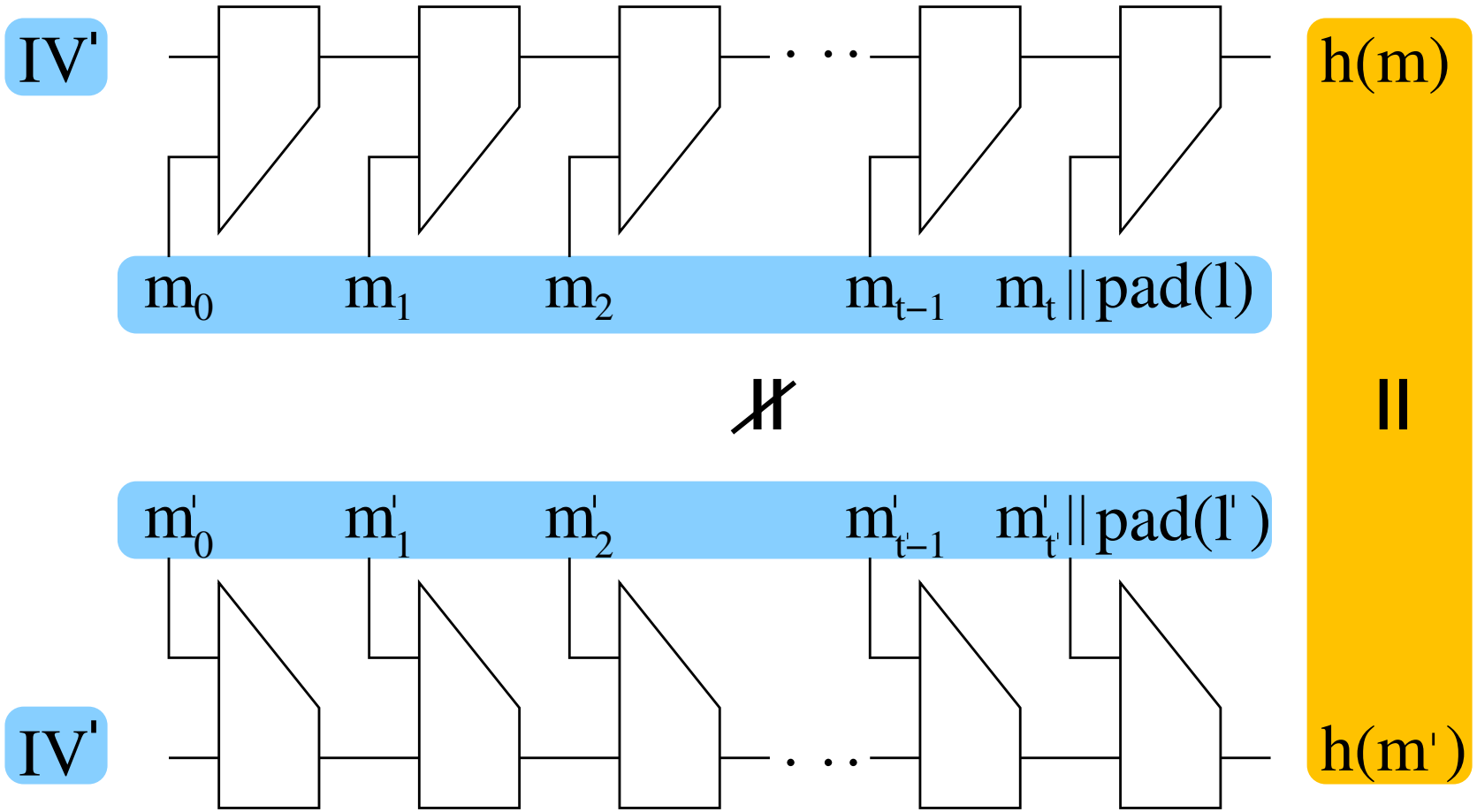
Collision Attacks on Iterated Hash Functions

Collision



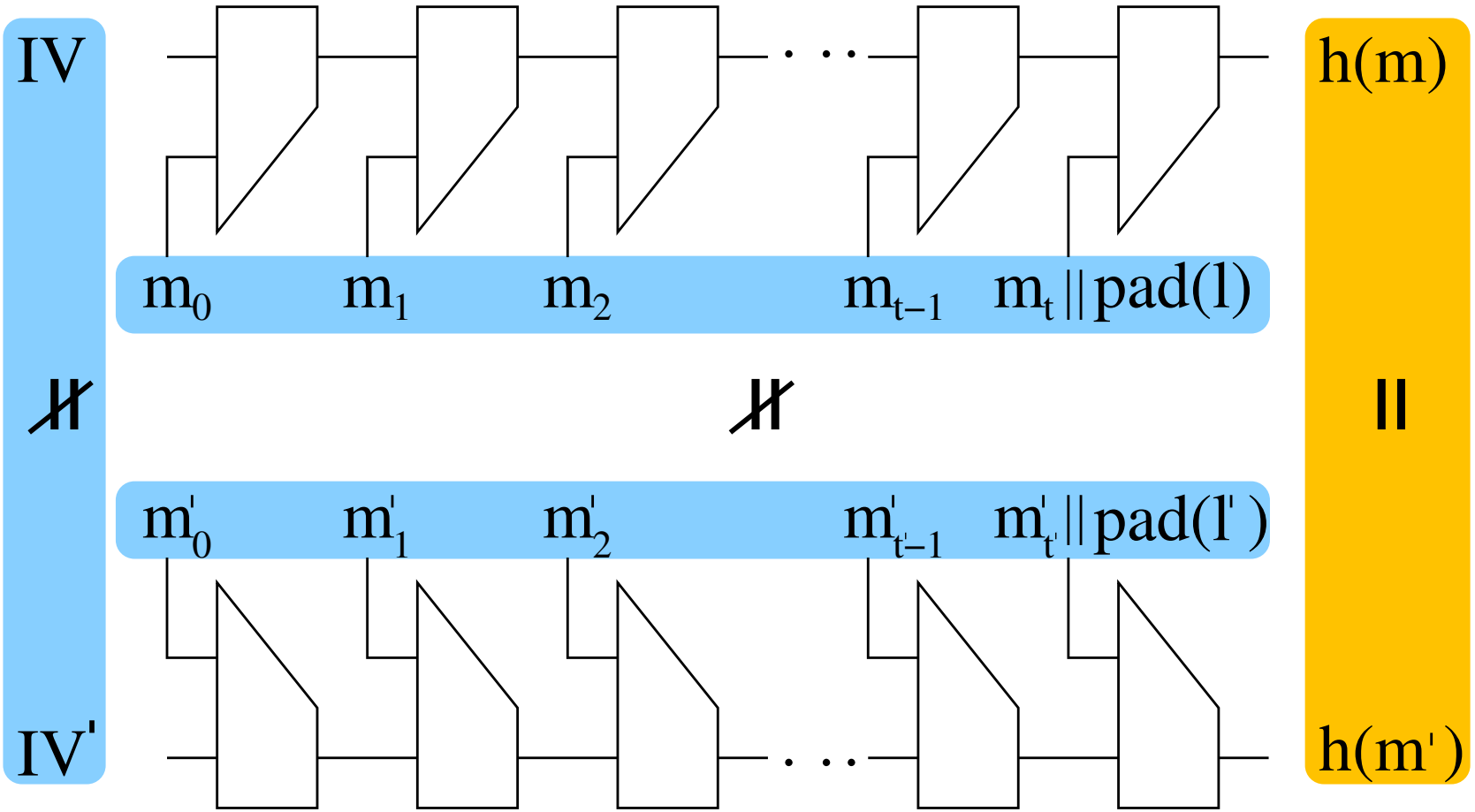
Collision Attacks on Iterated Hash Functions

Semi-free-start collision



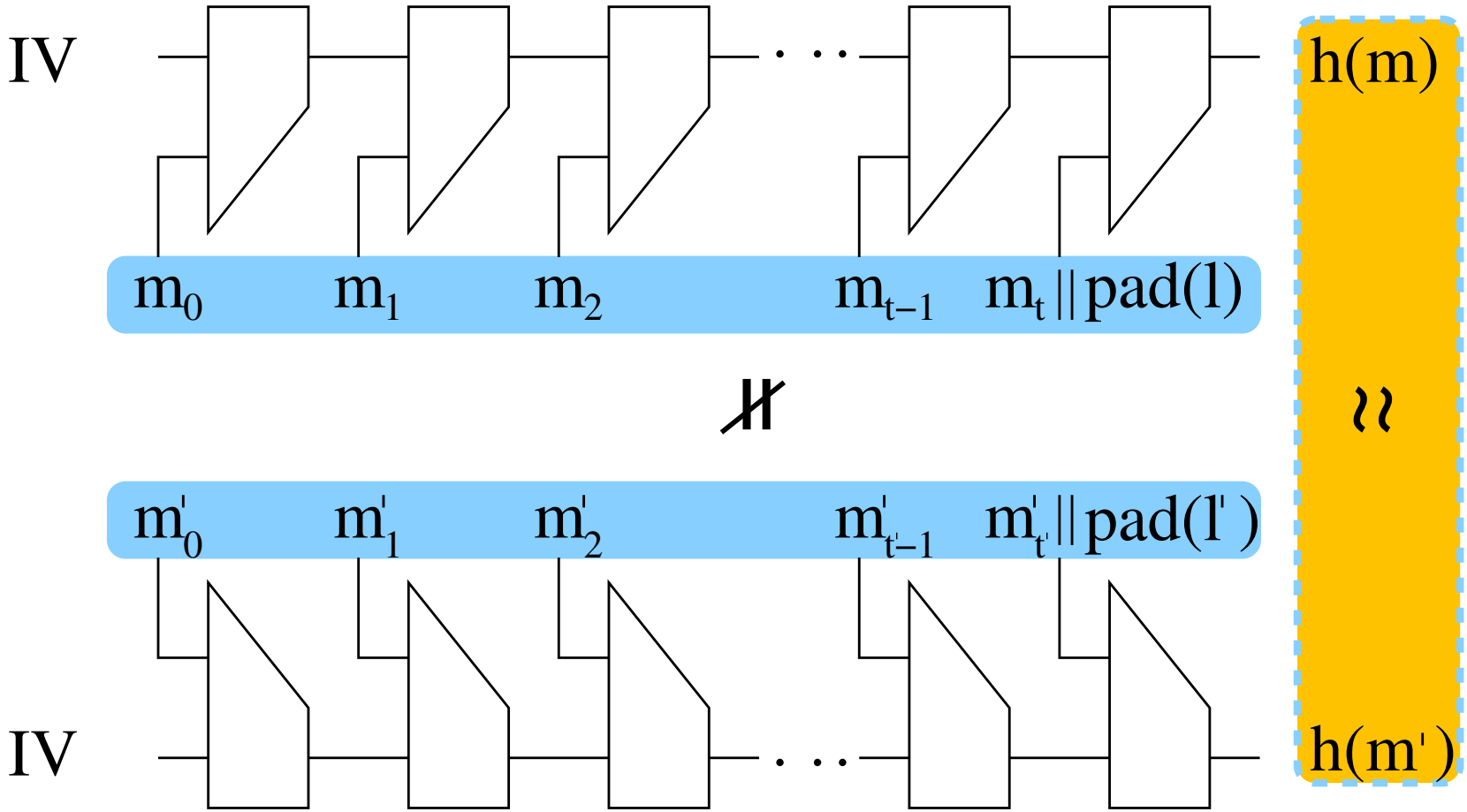
Collision Attacks on Iterated Hash Functions

Free-start collision

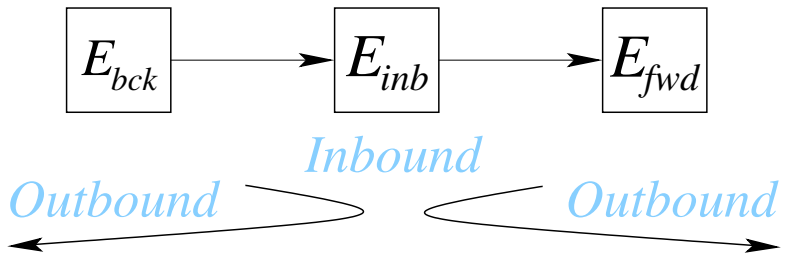


Collision Attacks on Iterated Hash Functions

Near-collision

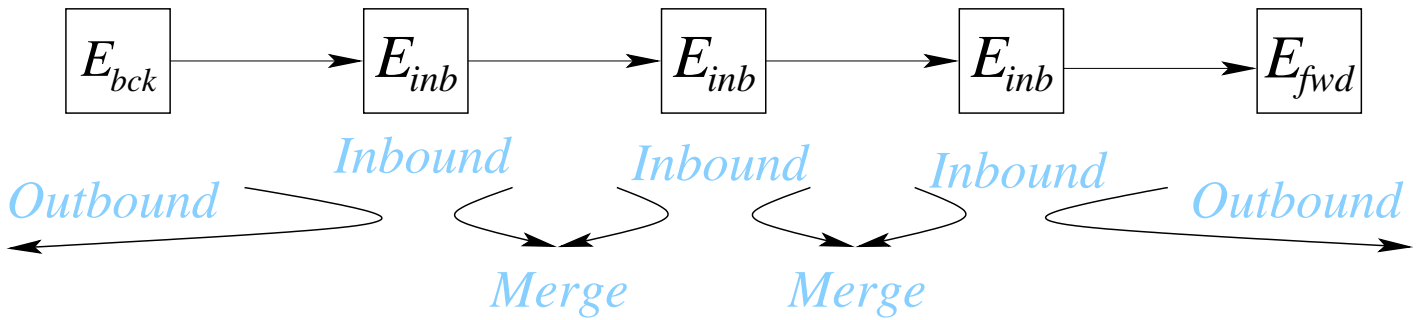


Rebound Attack



- Introduced by Mendel et al. in FSE 2009
- Combination of "Meet-in-the-Middle attack" and "Inside-Out approach".
- It consists of two main phases
  - Inbound Phase
  - Outbound Phase
- Applied to Whirlpool, Grøstl, LANE, ECHO ...

Improved Rebound Attack



- Lamberger et al. and Matusiewicz et al. in Asiacrypt 2009
- Mainly combination of "Meet-in-the-Middle attack" and "Inside-Out approach".
- It consists of three main phases
  - Inbound Phase
  - Merge Phase
  - Outbound Phase

Outline

JH

Designed by Wu

Second round SHA-3 candidate

Sponge based construction.

16.2 cycles/byte on 64-bit Core2 microprocessor

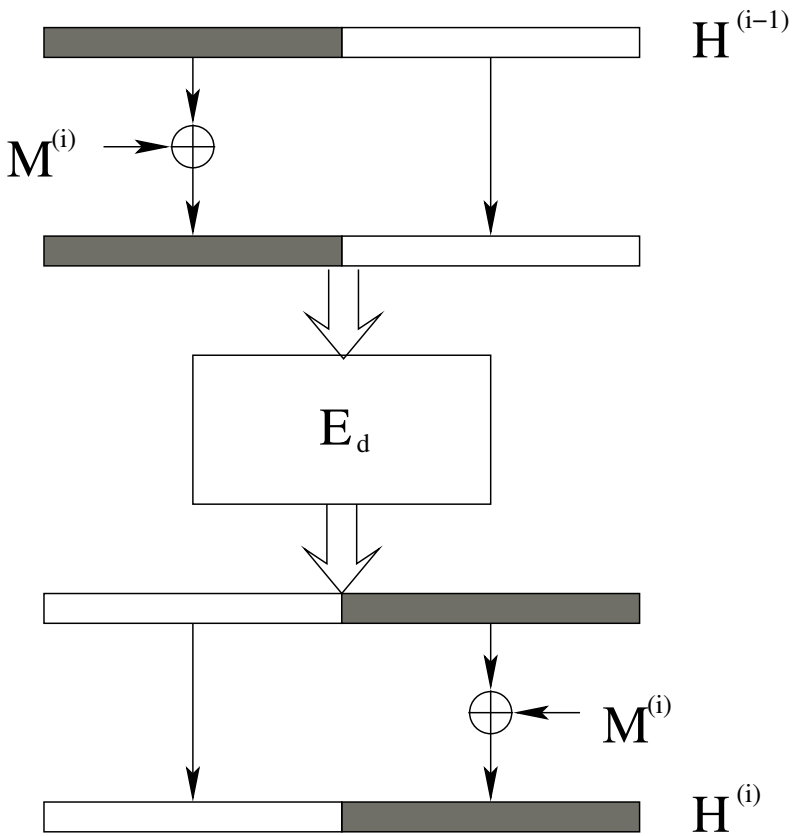
21.3 cycles/byte on 32-bit Core2 microprocessor

Our Contribution

Up to 16 round JH is not semi-free-start collision resistant.

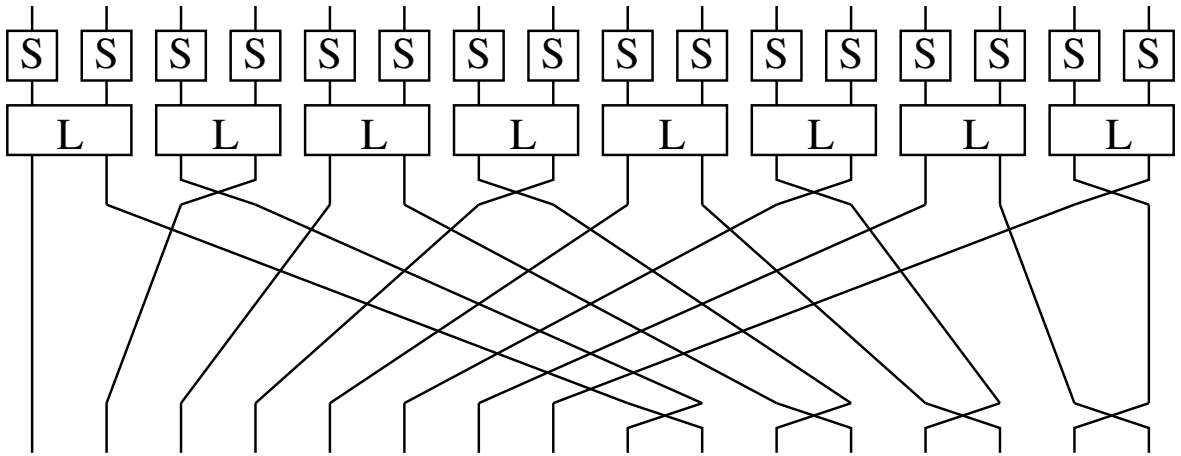
From 19 round to 22 round, semi-free-start near-collisions can be found for the compression function of JH.

JH



d	State Size	Message Block Size	# of Rounds
n	2^{n+2}	2^{n+1}	$5 \cdot (n-1) + 0.5$ $6 \cdot (n-1) + 0.5$
4	64	32	15.5
6	256	128	30.5
8	1024	512	35.5

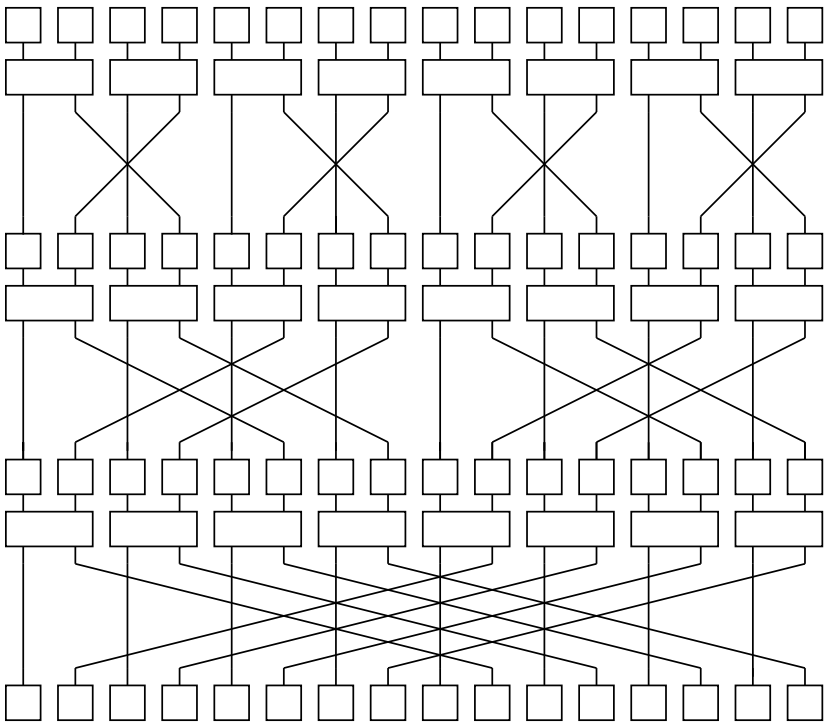
Round Function of JH



- 4-bit-to-4-bit S-Boxes (S_0-S_1)
- Linear Transformation based on $[4, 2, 3]$ MDS code
- Permutation P_d
- Round function with different round constants

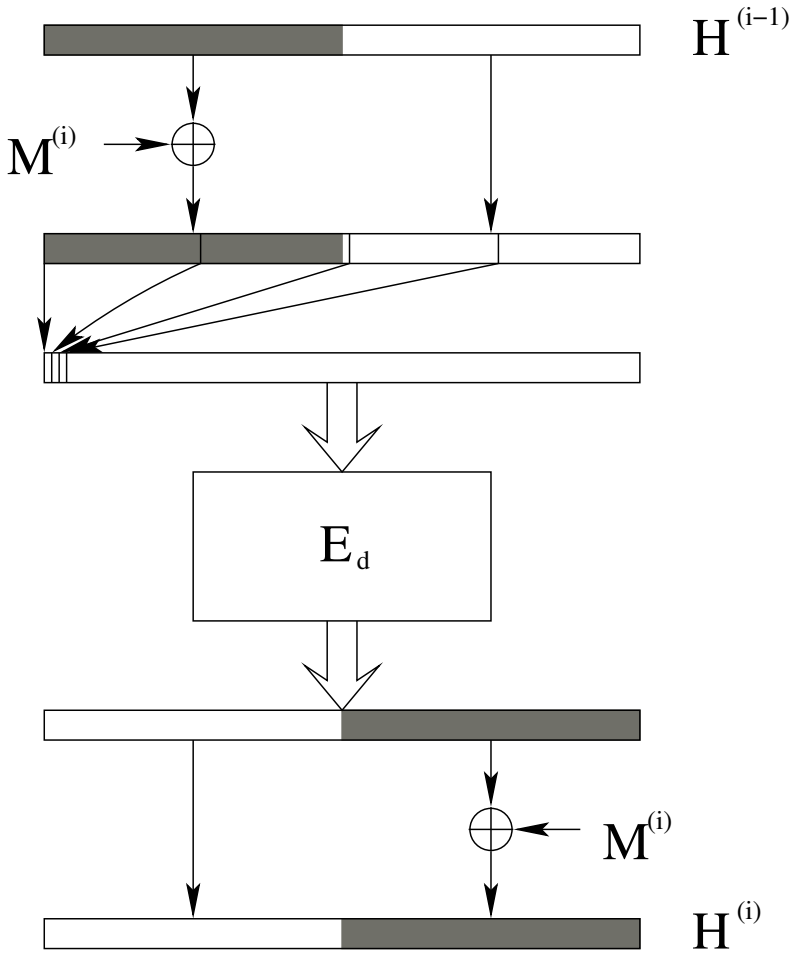
Bit-Slice Round Function of JH ($d = 4$)

Figure: 3 different round functions of bit slice



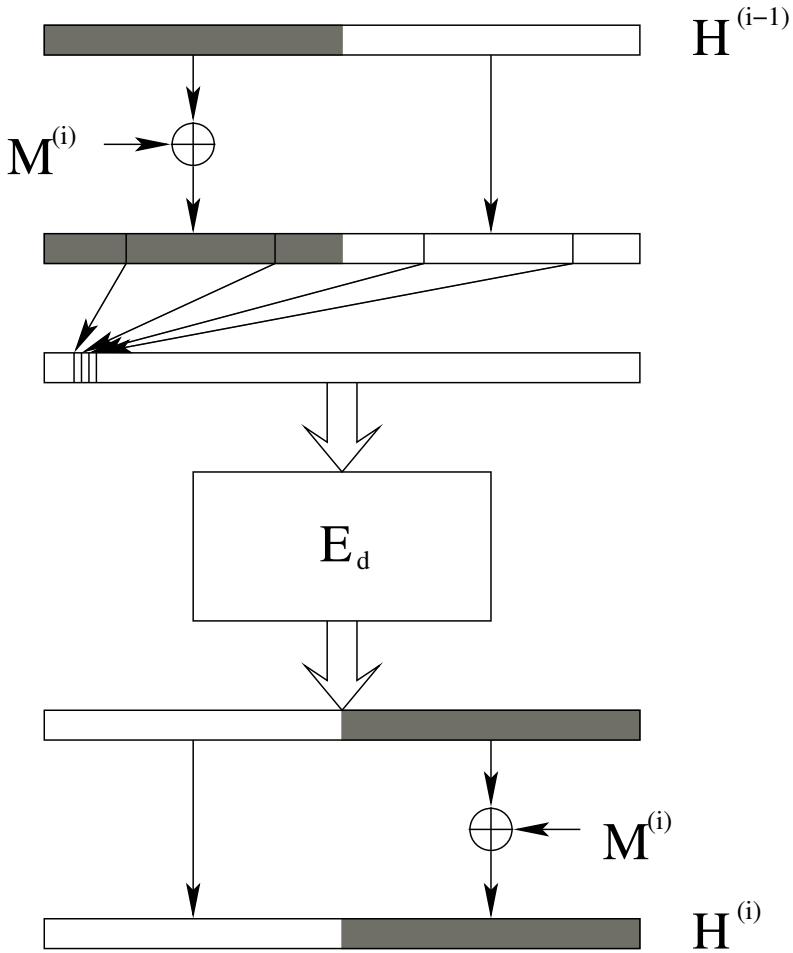
Grouping - De-grouping

Grouping



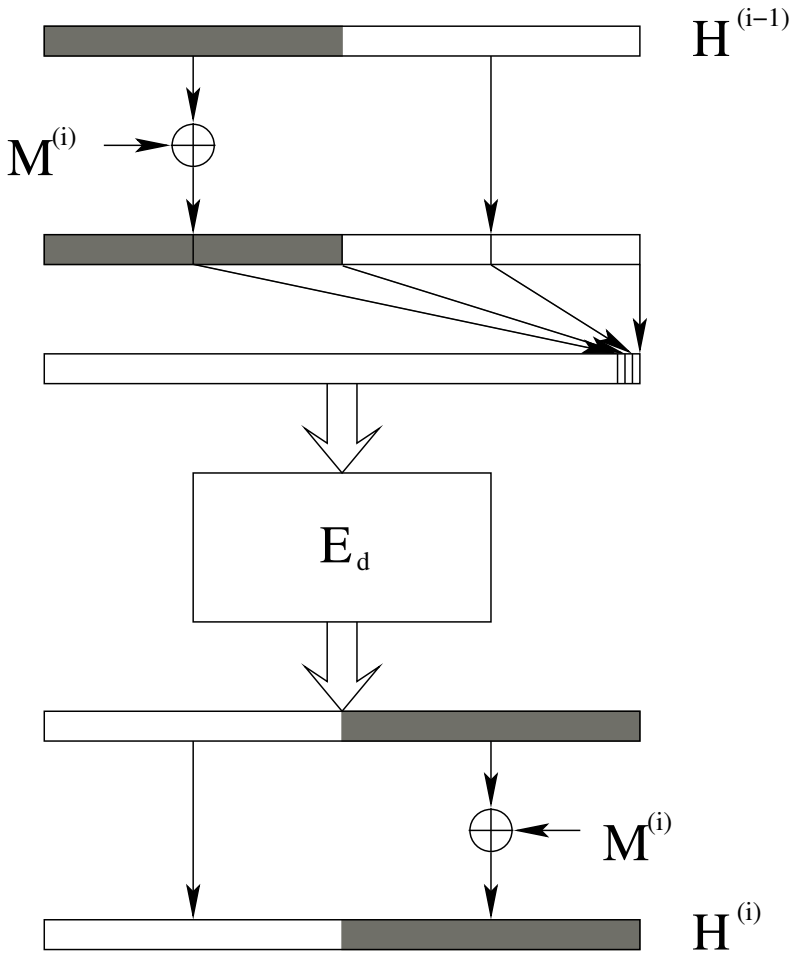
Grouping - De-grouping

Grouping



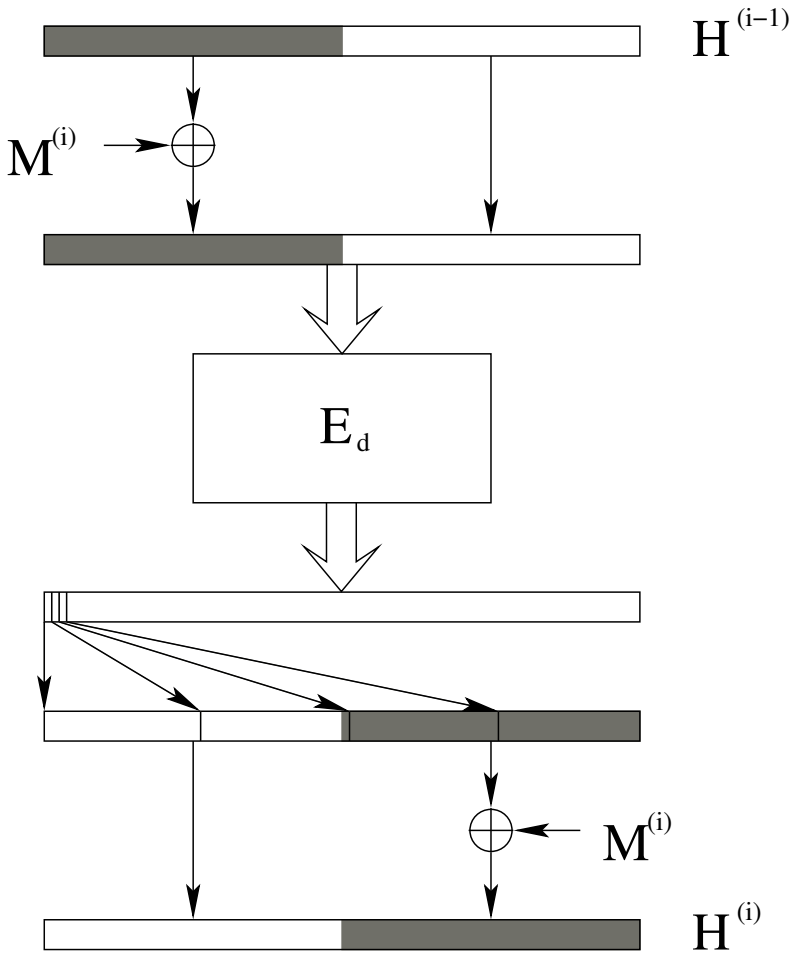
Grouping - De-grouping

Grouping



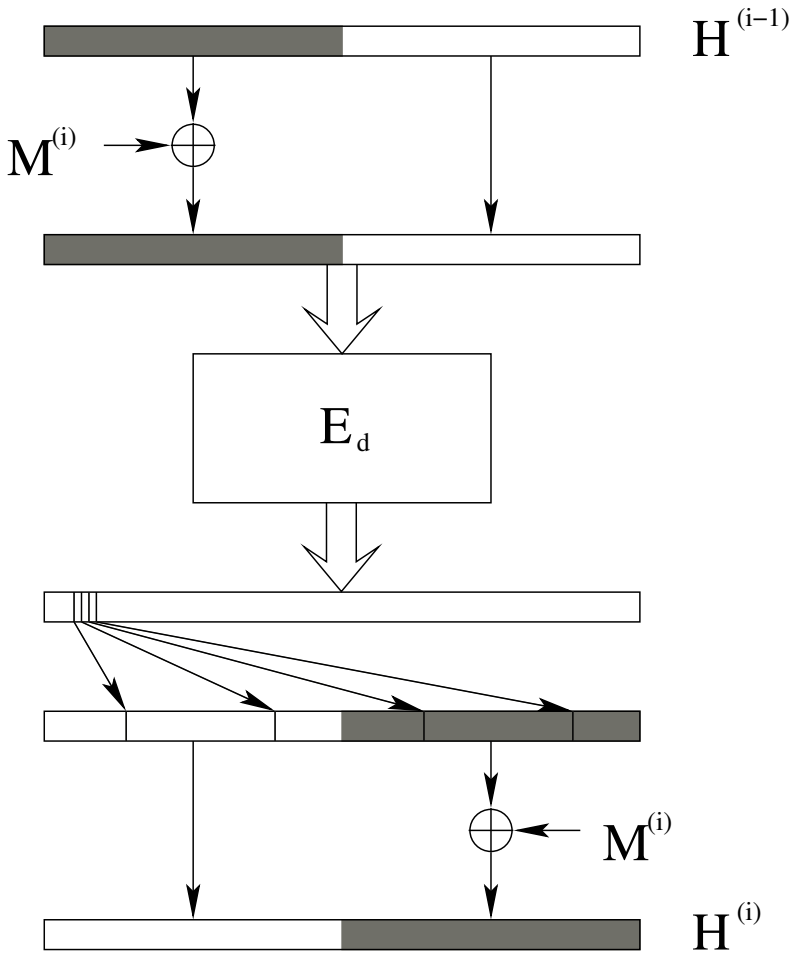
Grouping - De-grouping

DeGrouping



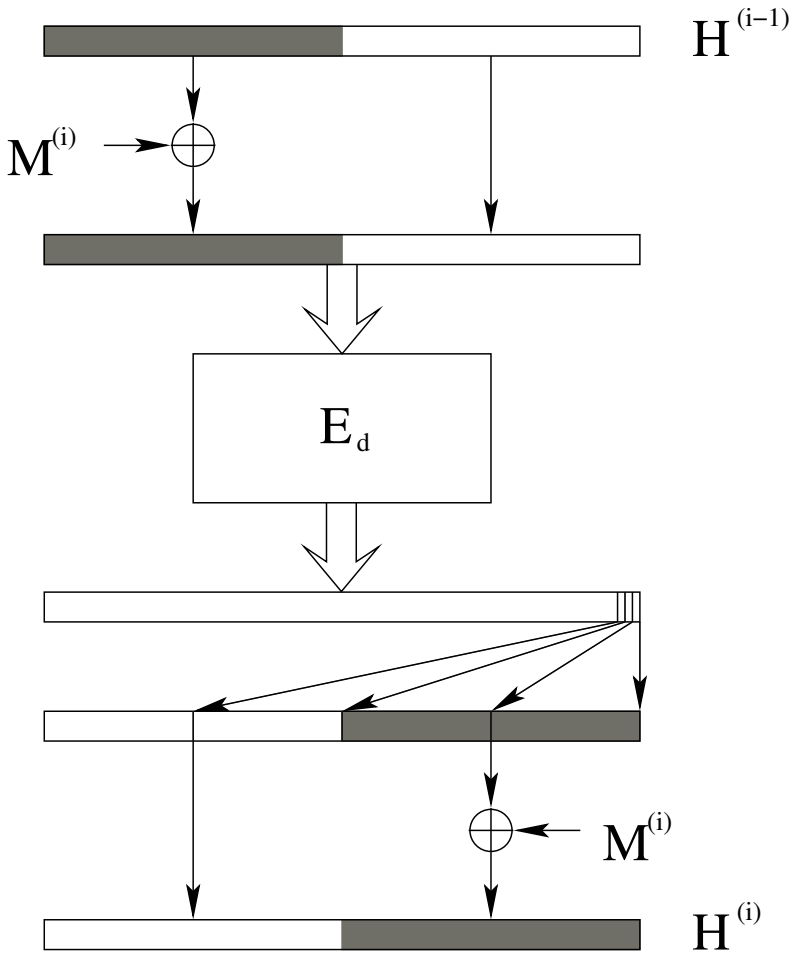
Grouping - De-grouping

DeGrouping



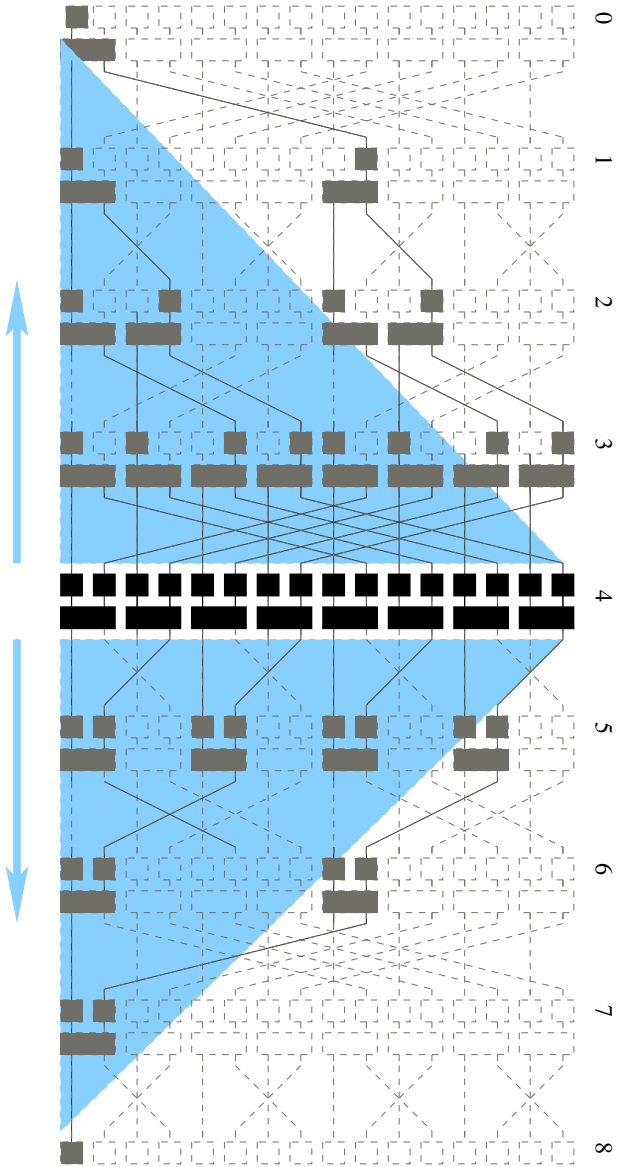
Grouping - De-grouping

DeGrouping



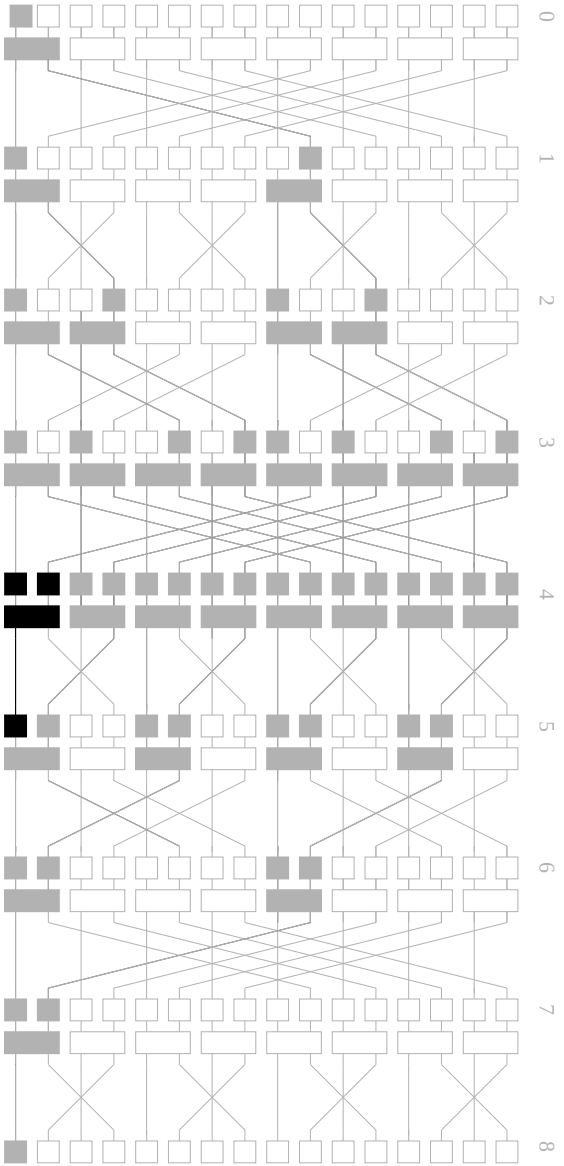
Outline

Overview of the Attack



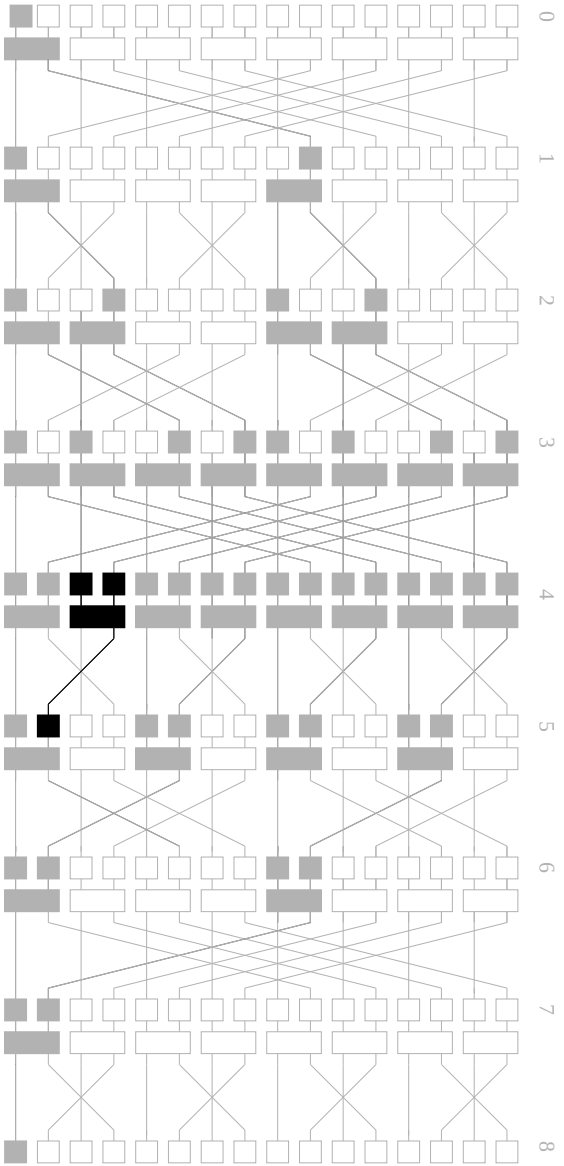
- Attack is first implemented to small version of JH $d = 4$.
- Same technique applied to the submitted version of JH with $d = 8$.
- Semi-free-start collisions found for reduced number of rounds.
- To improve the attack, a semi-automatic code is used.

Constructing Differential Path on Round Function



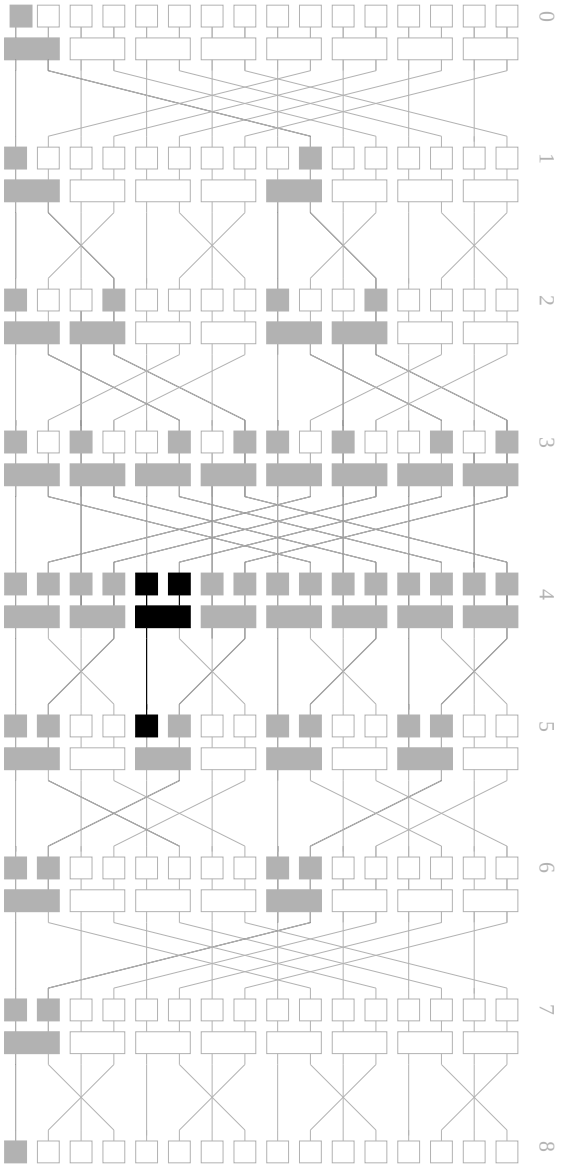
- Try all possible values, keep only the ones that satisfy the desired pattern

Constructing Differential Path on Round Function



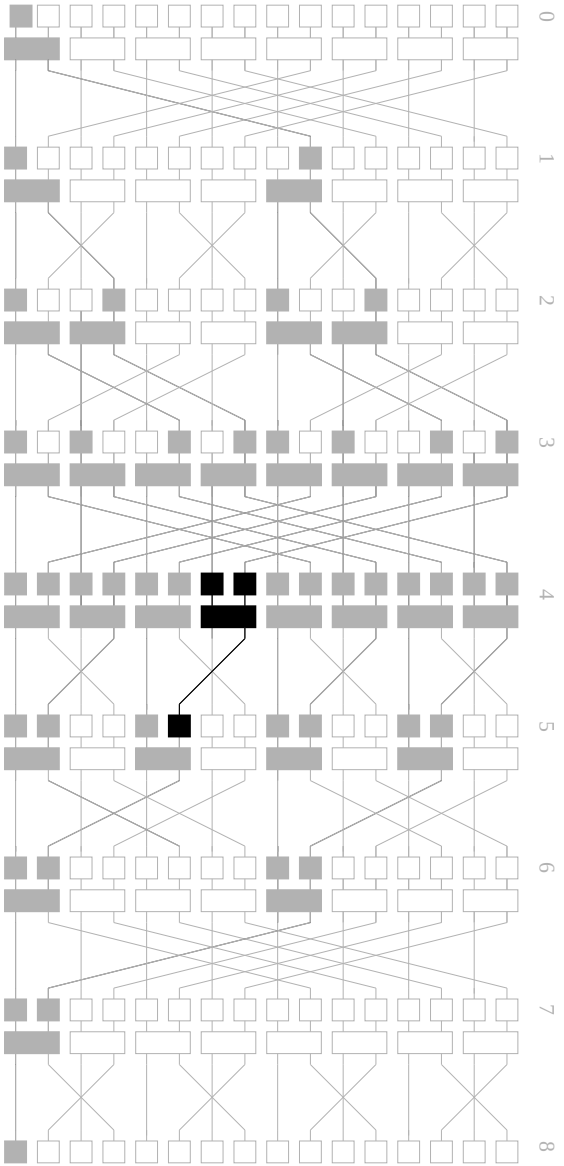
- Try all possible values, keep only the ones that satisfy the desired pattern

Constructing Differential Path on Round Function



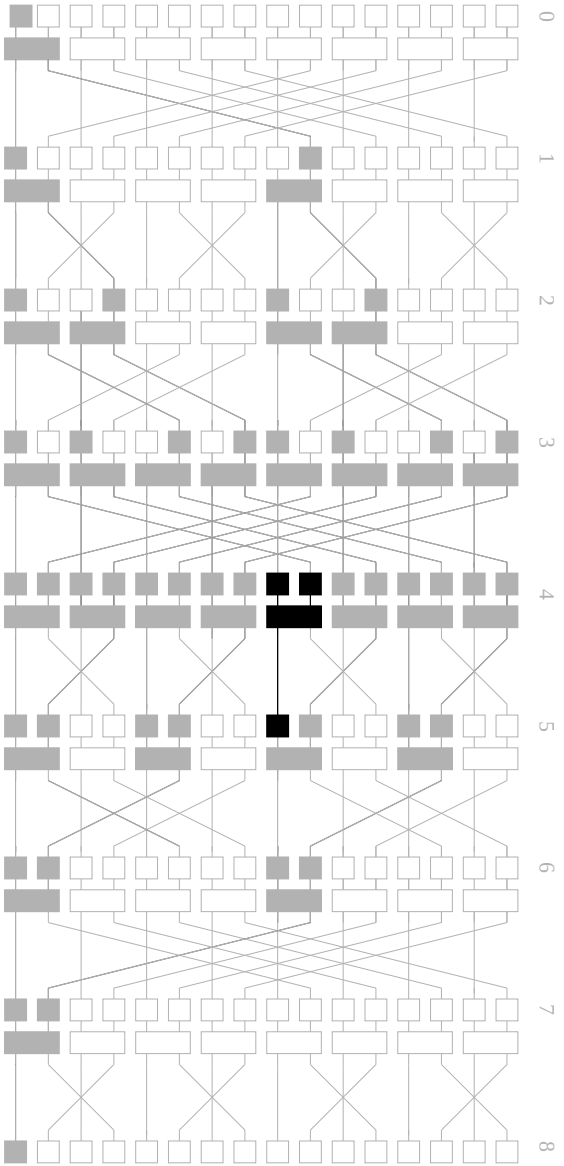
- Try all possible values, keep only the ones that satisfy the desired pattern

Constructing Differential Path on Round Function



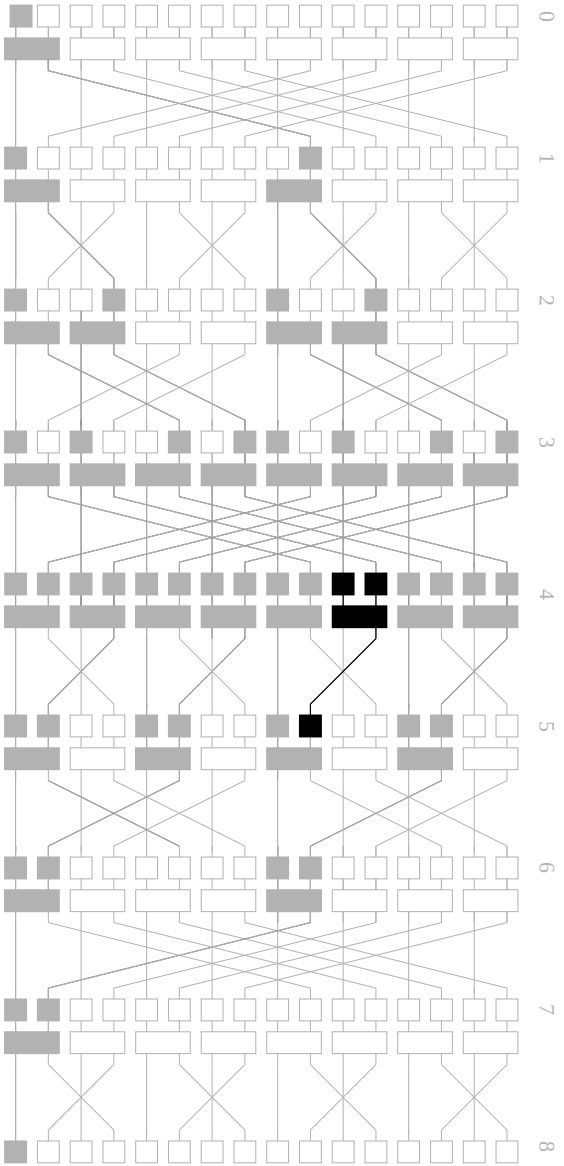
- Try all possible values, keep only the ones that satisfy the desired pattern

Constructing Differential Path on Round Function



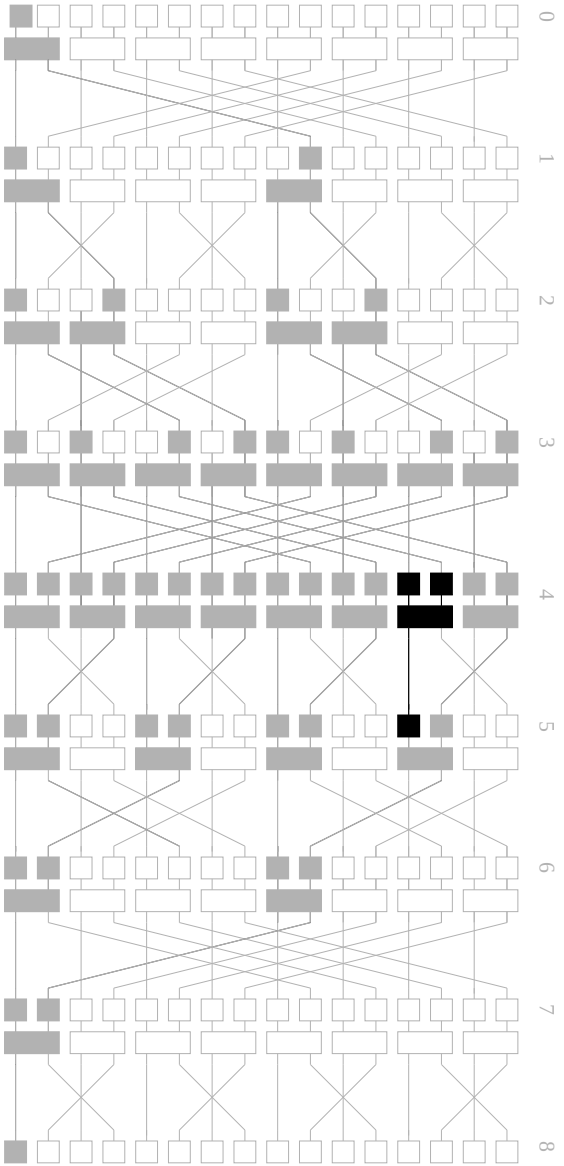
- Try all possible values, keep only the ones that satisfy the desired pattern

Constructing Differential Path on Round Function



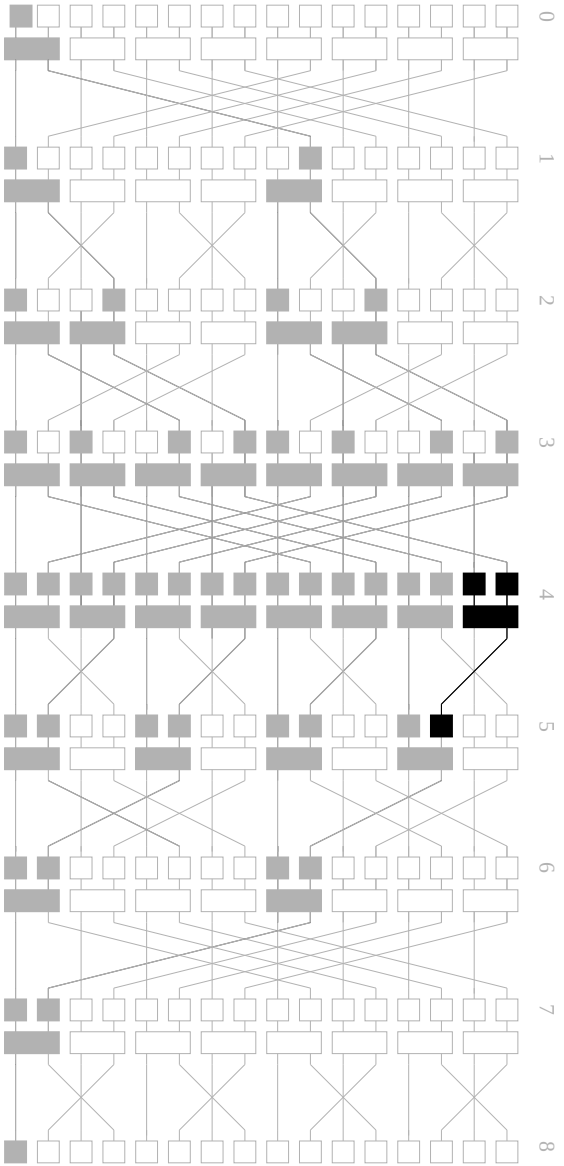
- Try all possible values, keep only the ones that satisfy the desired pattern

Constructing Differential Path on Round Function



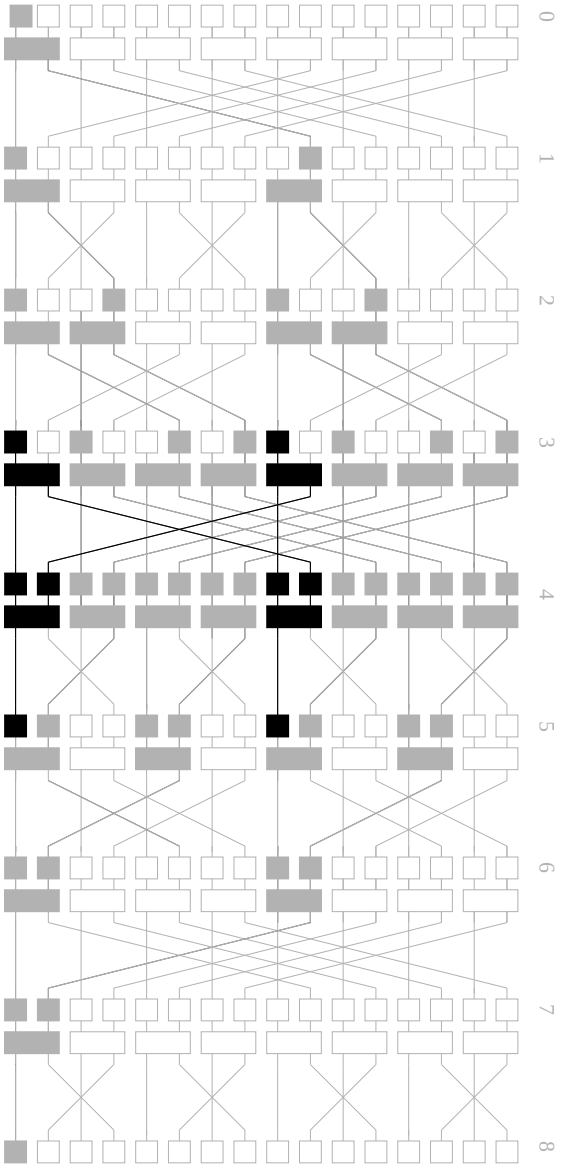
- Try all possible values, keep only the ones that satisfy the desired pattern

Constructing Differential Path on Round Function



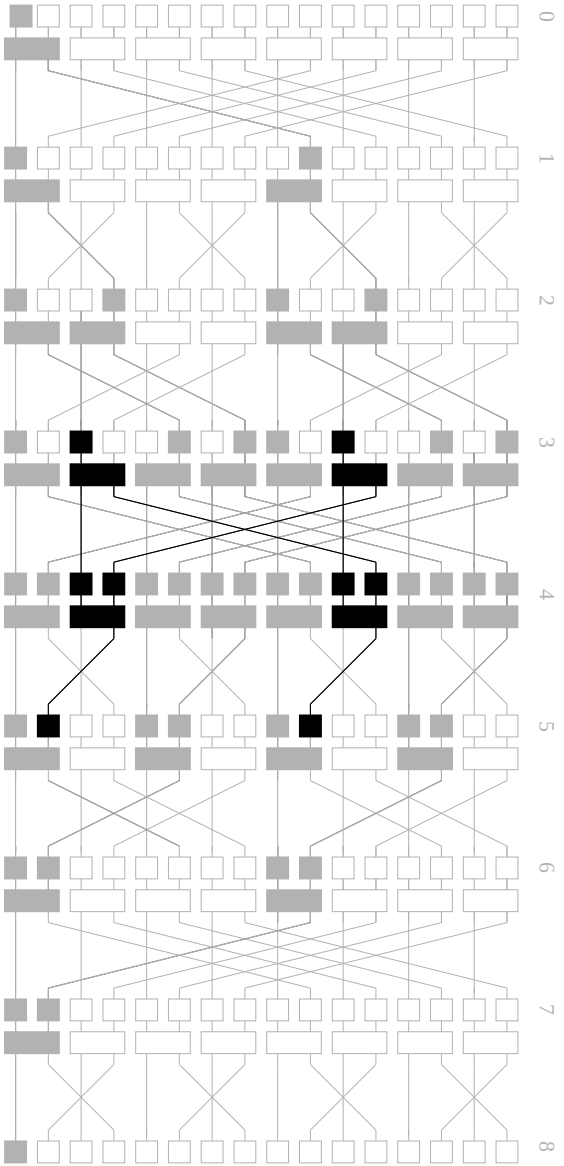
- Try all possible values, keep only the ones that satisfy the desired pattern

Constructing Differential Path on Round Function



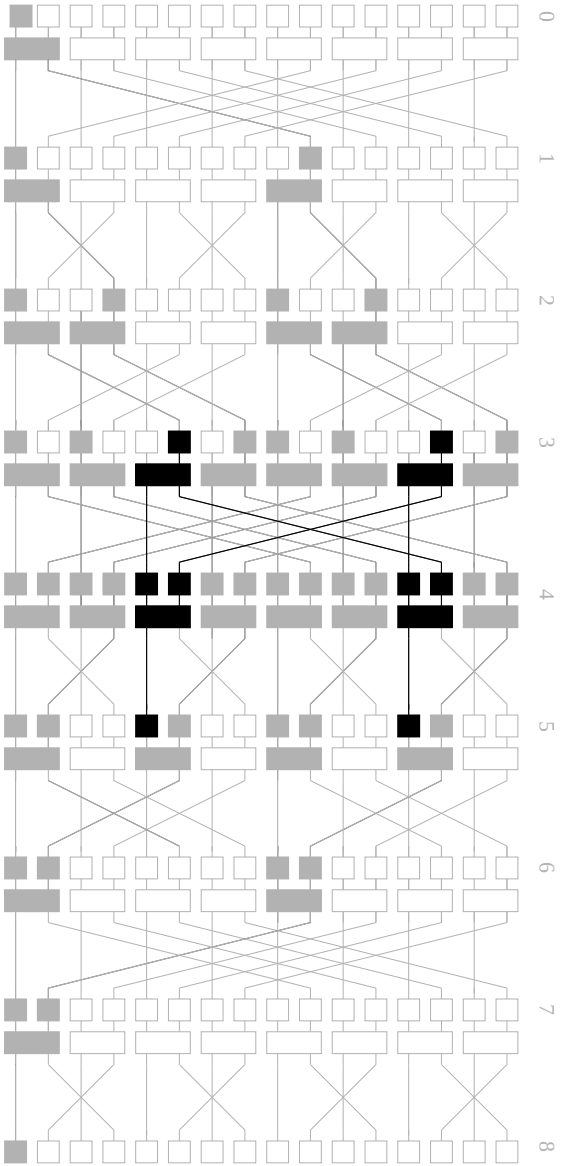
- Try all possible values, keep only the ones that satisfy the desired pattern
- Compute the cross-product of the highlighted sets and check if the differences satisfy the filtering conditions.

Constructing Differential Path on Round Function



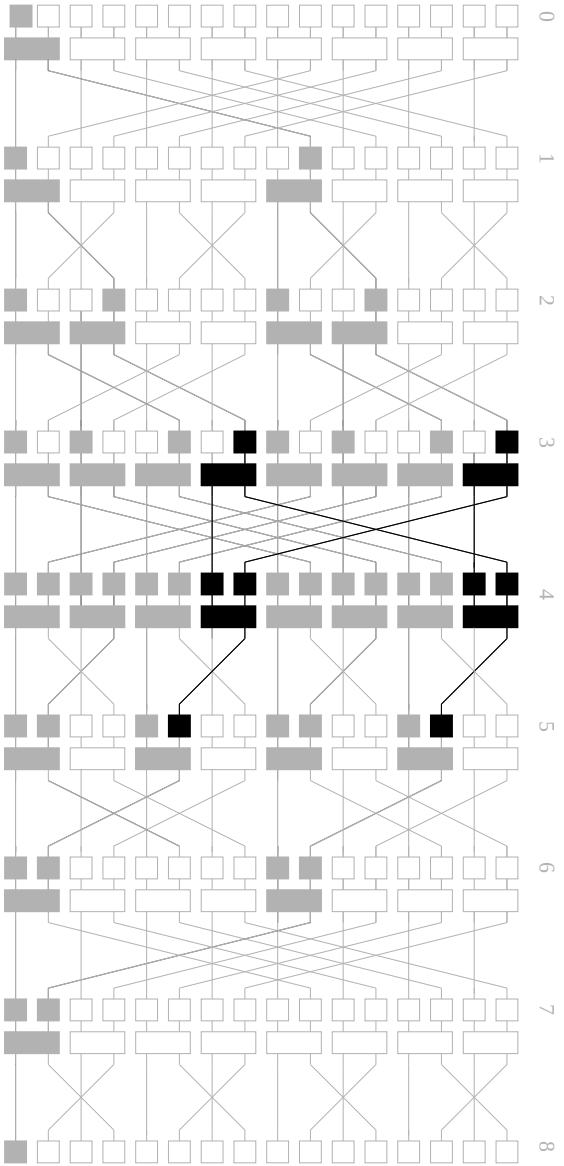
- Try all possible values, keep only the ones that satisfy the desired pattern
- Compute the cross-product of the highlighted sets and check if the differences satisfy the filtering conditions.

Constructing Differential Path on Round Function



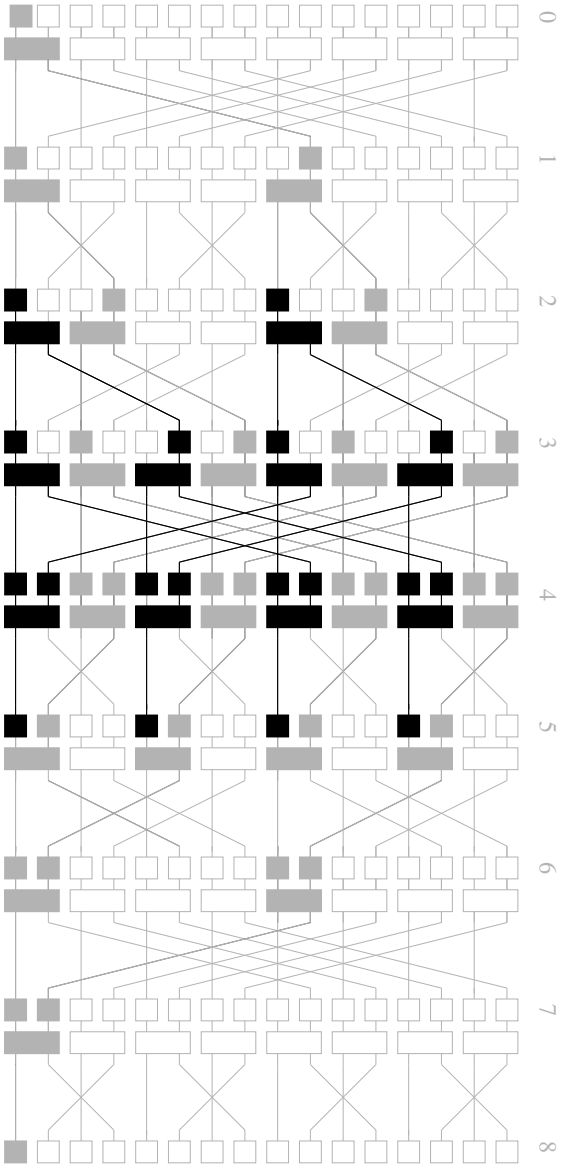
- Try all possible values, keep only the ones that satisfy the desired pattern
- Compute the cross-product of the highlighted sets and check if the differences satisfy the filtering conditions.

Constructing Differential Path on Round Function



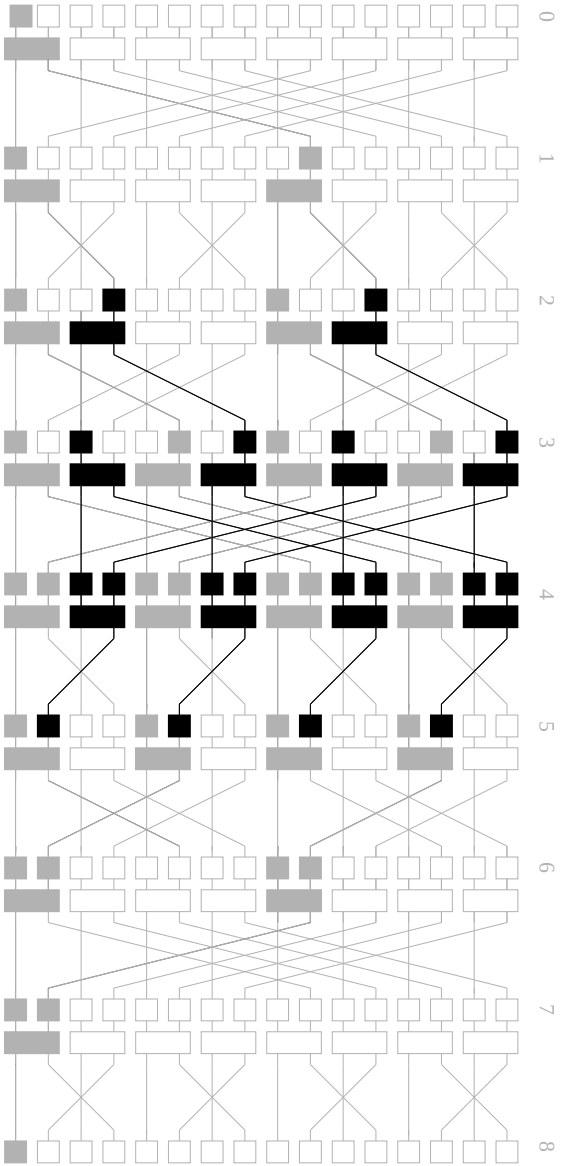
- Try all possible values, keep only the ones that satisfy the desired pattern
- Compute the cross-product of the highlighted sets and check if the differences satisfy the filtering conditions.

Constructing Differential Path on Round Function



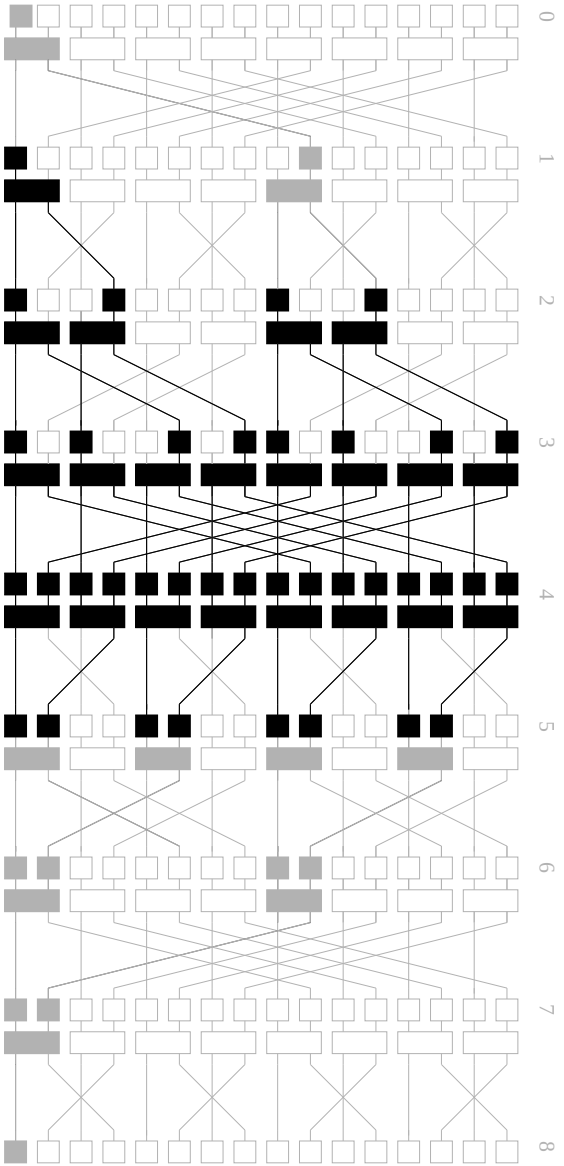
- Try all possible values, keep only the ones that satisfy the desired pattern
- Compute the cross-product of the highlighted sets and check if the differences satisfy the filtering conditions.
- Expand the sets by computing the cross-product of the highlighted sets.

Constructing Differential Path on Round Function



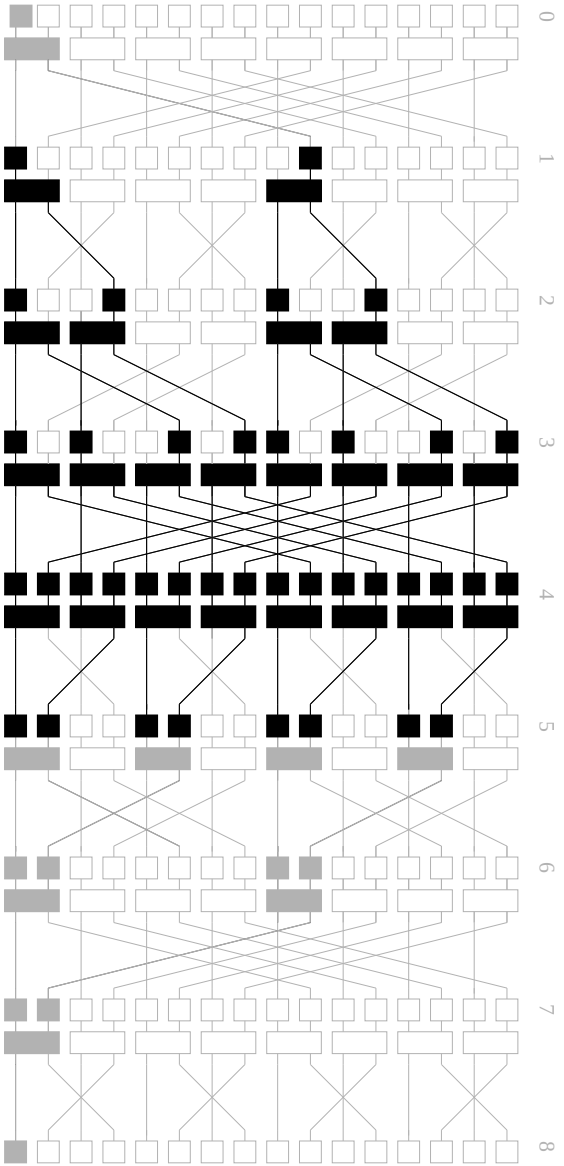
- Try all possible values, keep only the ones that satisfy the desired pattern
- Compute the cross-product of the highlighted sets and check if the differences satisfy the filtering conditions.
- Expand the sets by computing the cross-product of the highlighted sets.

Constructing Differential Path on Round Function



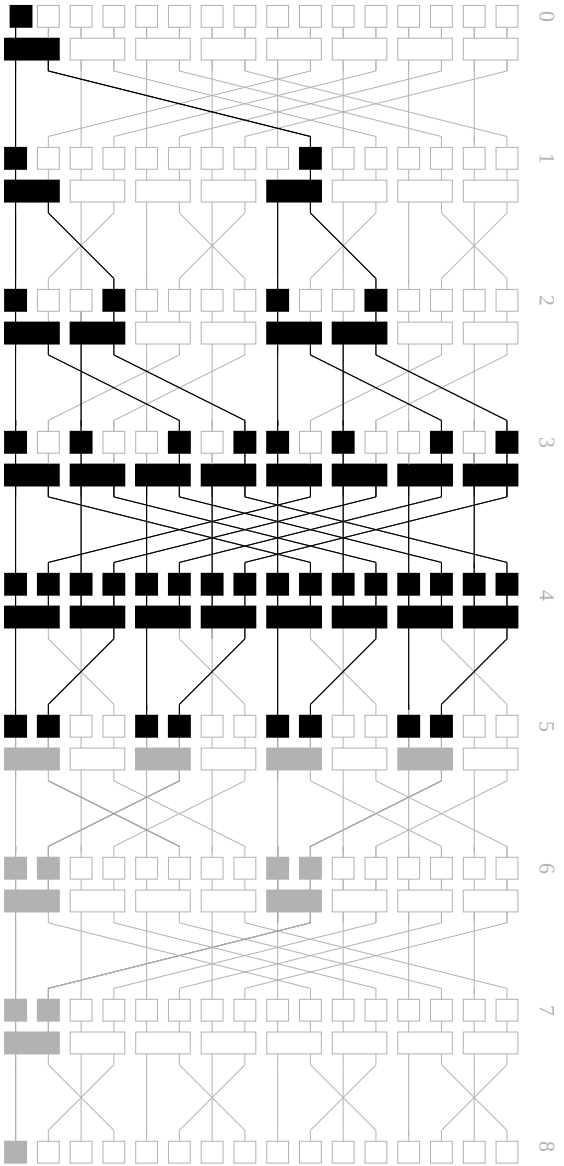
- Try all possible values, keep only the ones that satisfy the desired pattern
- Compute the cross-product of the highlighted sets and check if the differences satisfy the filtering conditions.
- Expand the sets by computing the cross-product of the highlighted sets.
- Compute the cross-product of the last two sets and check whether remaining 10 filtering conditions are satisfied or not.

Constructing Differential Path on Round Function



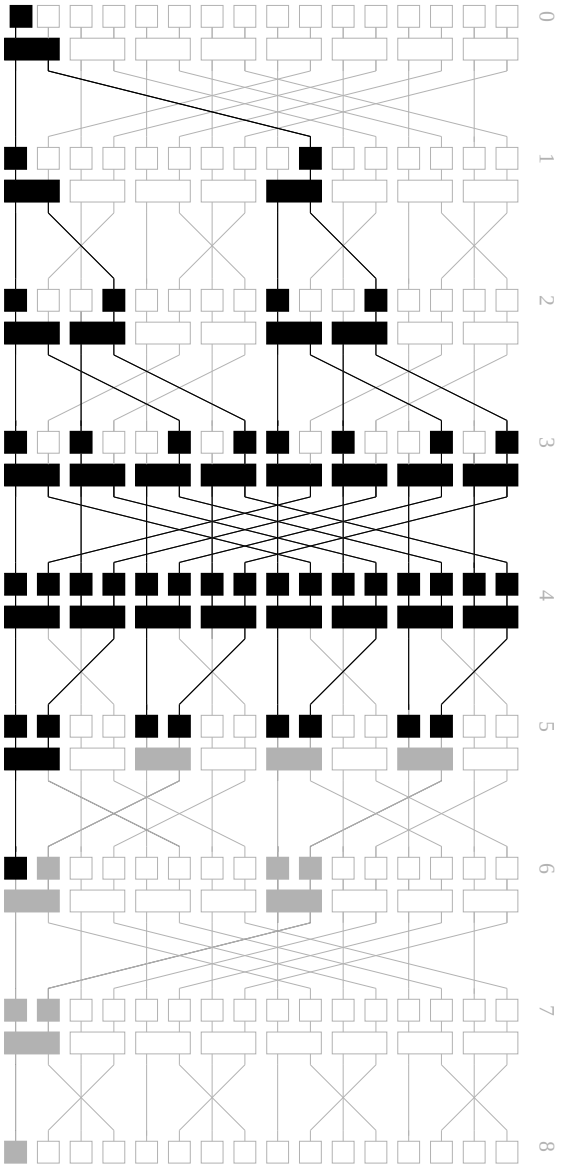
- Try all possible values, keep only the ones that satisfy the desired pattern
- Compute the cross-product of the highlighted sets and check if the differences satisfy the filtering conditions.
- Expand the sets by computing the cross-product of the highlighted sets.
- Compute the cross-product of the last two sets and check whether remaining 10 filtering conditions are satisfied or not.

Constructing Differential Path on Round Function



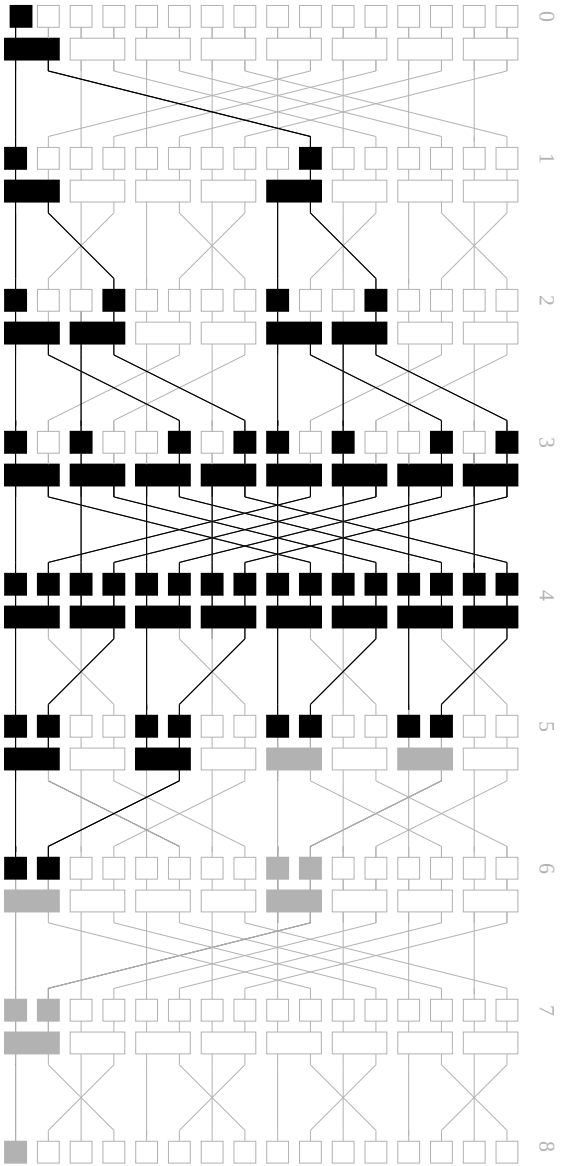
- Try all possible values, keep only the ones that satisfy the desired pattern
- Compute the cross-product of the highlighted sets and check if the differences satisfy the filtering conditions.
- Expand the sets by computing the cross-product of the highlighted sets.
- Compute the cross-product of the last two sets and check whether remaining 10 filtering conditions are satisfied or not.

Constructing Differential Path on Round Function



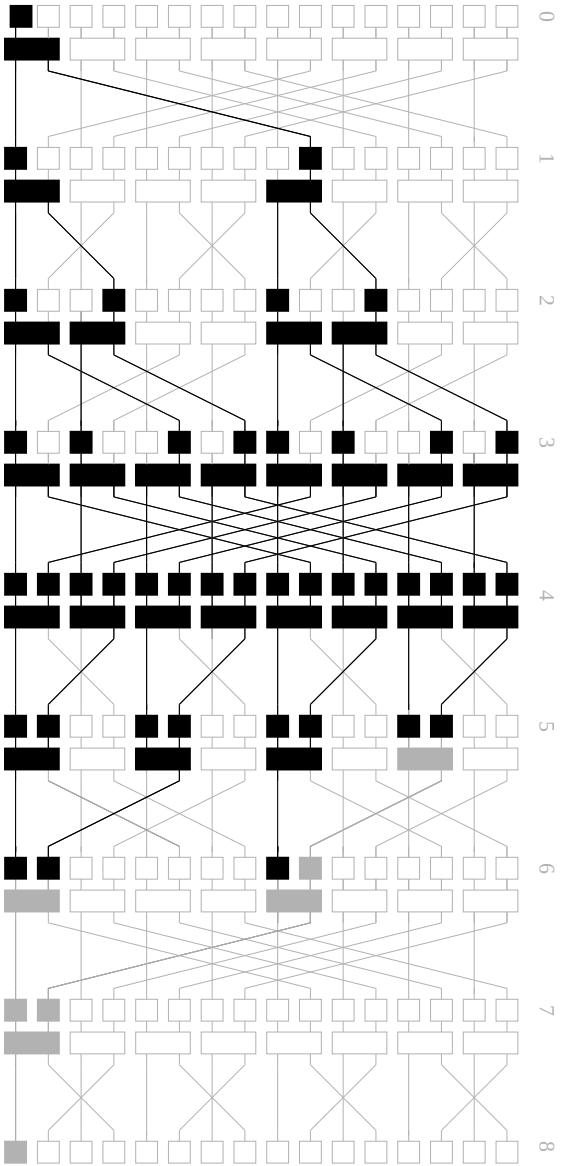
- Try all possible values, keep only the ones that satisfy the desired pattern
- Compute the cross-product of the highlighted sets and check if the differences satisfy the filtering conditions.
- Expand the sets by computing the cross-product of the highlighted sets.
- Compute the cross-product of the last two sets and check whether remaining 10 filtering conditions are satisfied or not.

Constructing Differential Path on Round Function



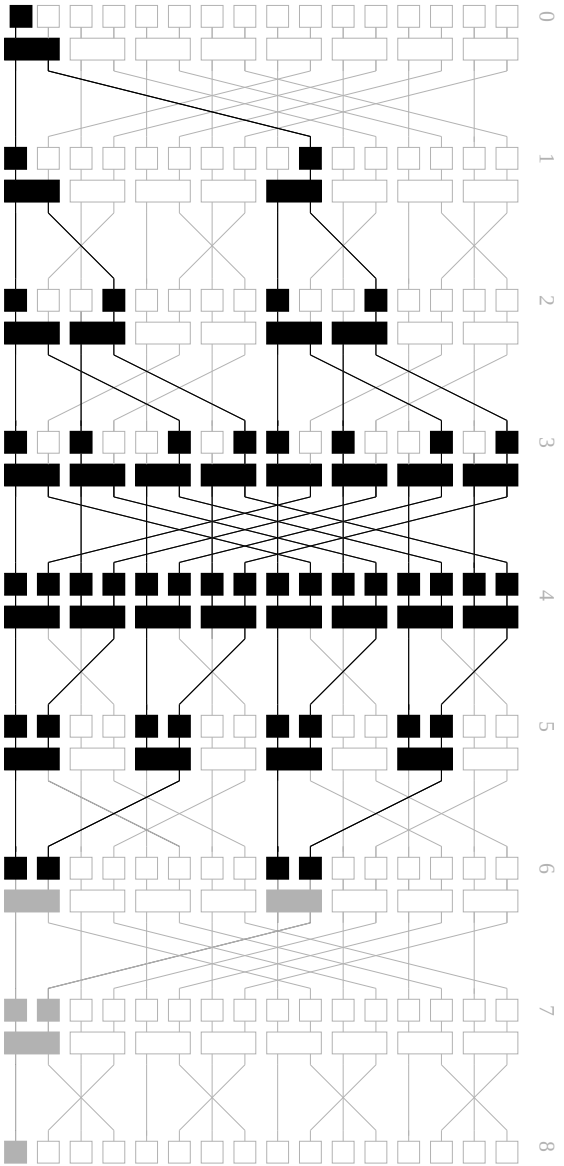
- Try all possible values, keep only the ones that satisfy the desired pattern
- Compute the cross-product of the highlighted sets and check if the differences satisfy the filtering conditions.
- Expand the sets by computing the cross-product of the highlighted sets.
- Compute the cross-product of the last two sets and check whether remaining 10 filtering conditions are satisfied or not.

Constructing Differential Path on Round Function



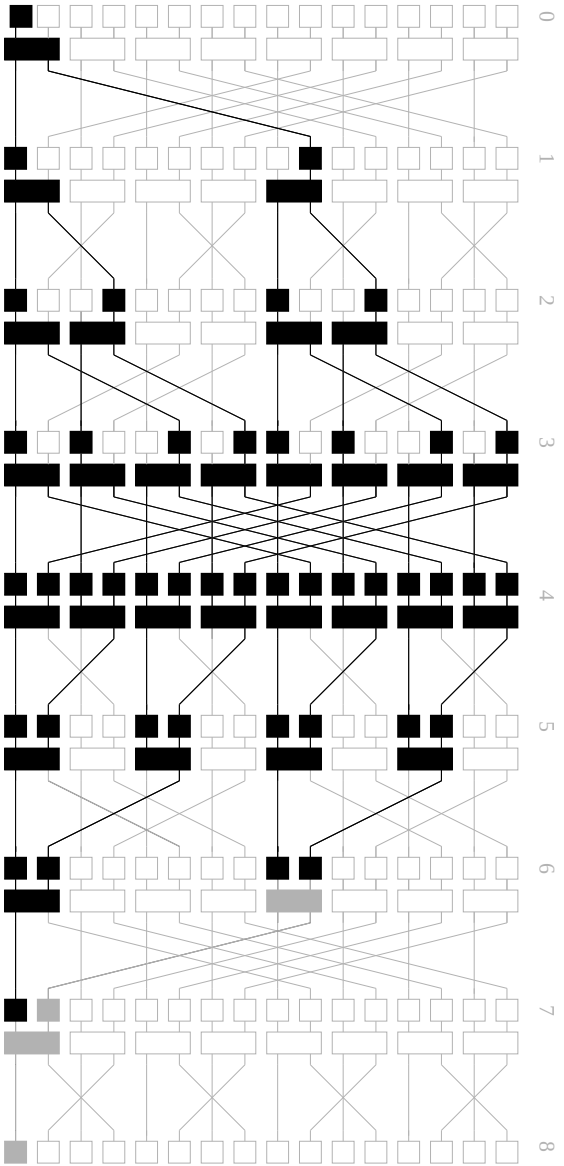
- Try all possible values, keep only the ones that satisfy the desired pattern
- Compute the cross-product of the highlighted sets and check if the differences satisfy the filtering conditions.
- Expand the sets by computing the cross-product of the highlighted sets.
- Compute the cross-product of the last two sets and check whether remaining 10 filtering conditions are satisfied or not.

Constructing Differential Path on Round Function



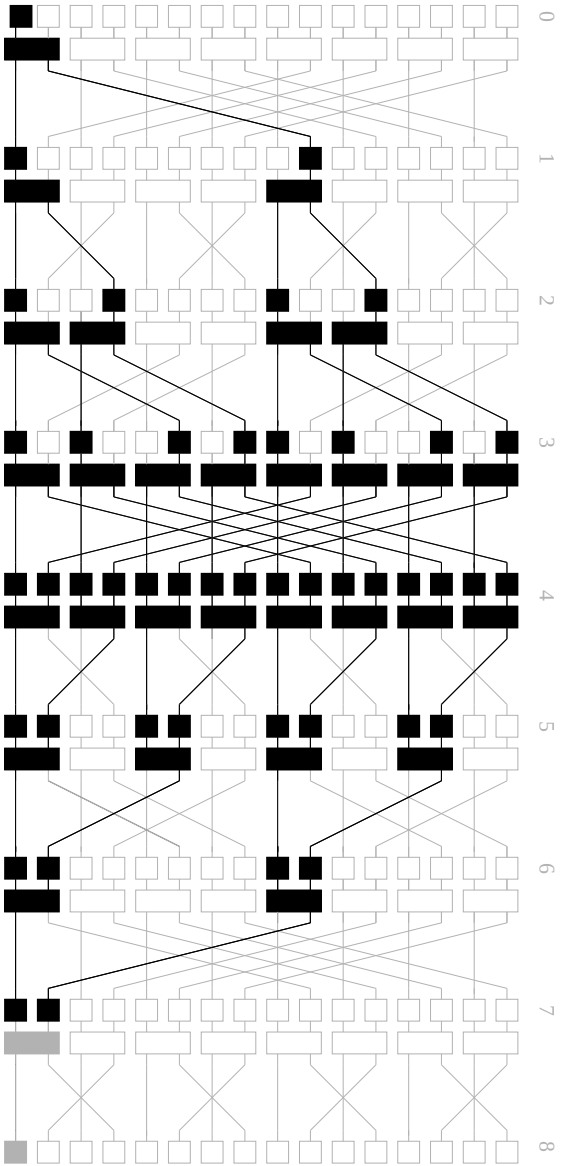
- Try all possible values, keep only the ones that satisfy the desired pattern
- Compute the cross-product of the highlighted sets and check if the differences satisfy the filtering conditions.
- Expand the sets by computing the cross-product of the highlighted sets.
- Compute the cross-product of the last two sets and check whether remaining 10 filtering conditions are satisfied or not.

Constructing Differential Path on Round Function



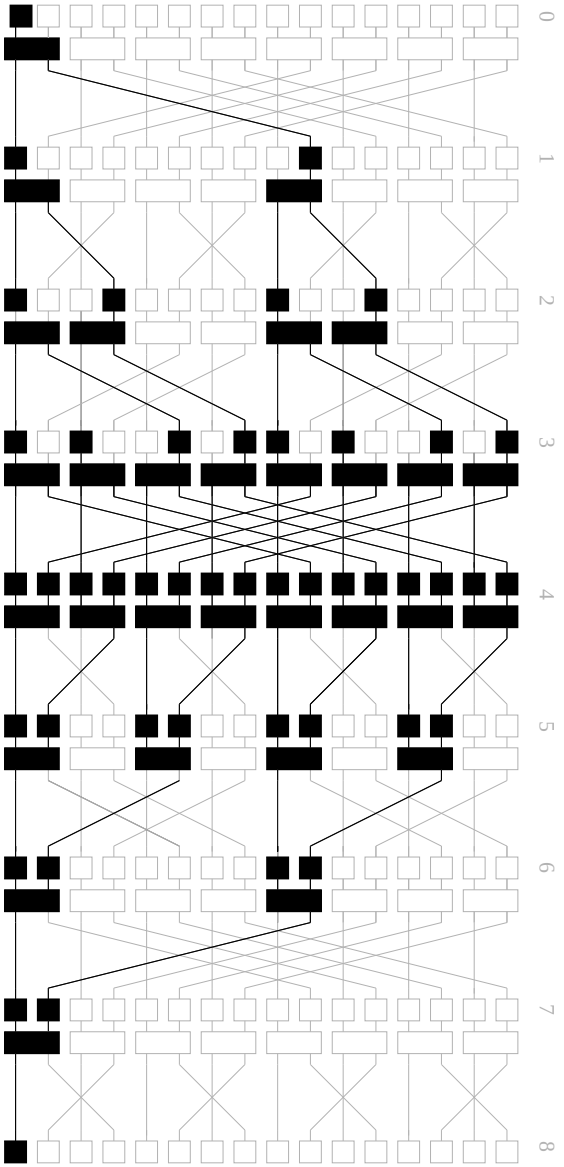
- Try all possible values, keep only the ones that satisfy the desired pattern
- Compute the cross-product of the highlighted sets and check if the differences satisfy the filtering conditions.
- Expand the sets by computing the cross-product of the highlighted sets.
- Compute the cross-product of the last two sets and check whether remaining 10 filtering conditions are satisfied or not.

Constructing Differential Path on Round Function



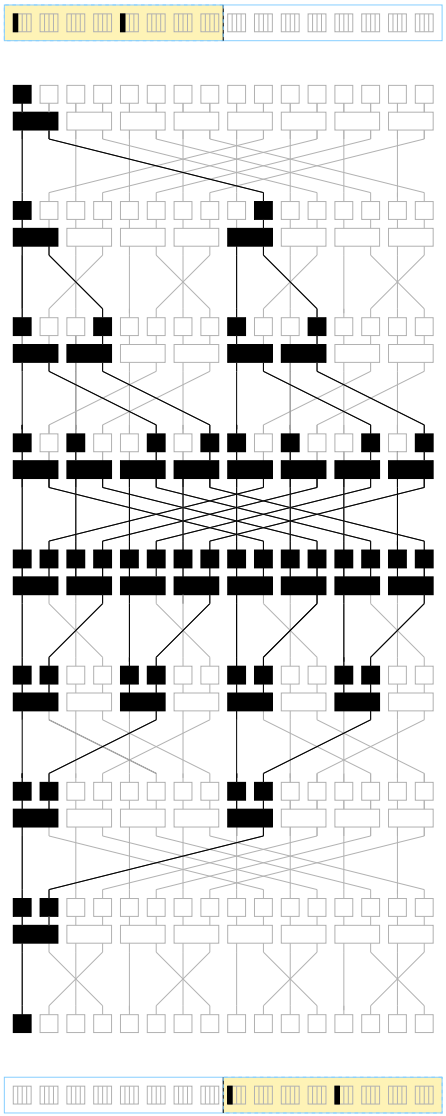
- Try all possible values, keep only the ones that satisfy the desired pattern
- Compute the cross-product of the highlighted sets and check if the differences satisfy the filtering conditions.
- Expand the sets by computing the cross-product of the highlighted sets.
- Compute the cross-product of the last two sets and check whether remaining 10 filtering conditions are satisfied or not.

Constructing Differential Path on Round Function



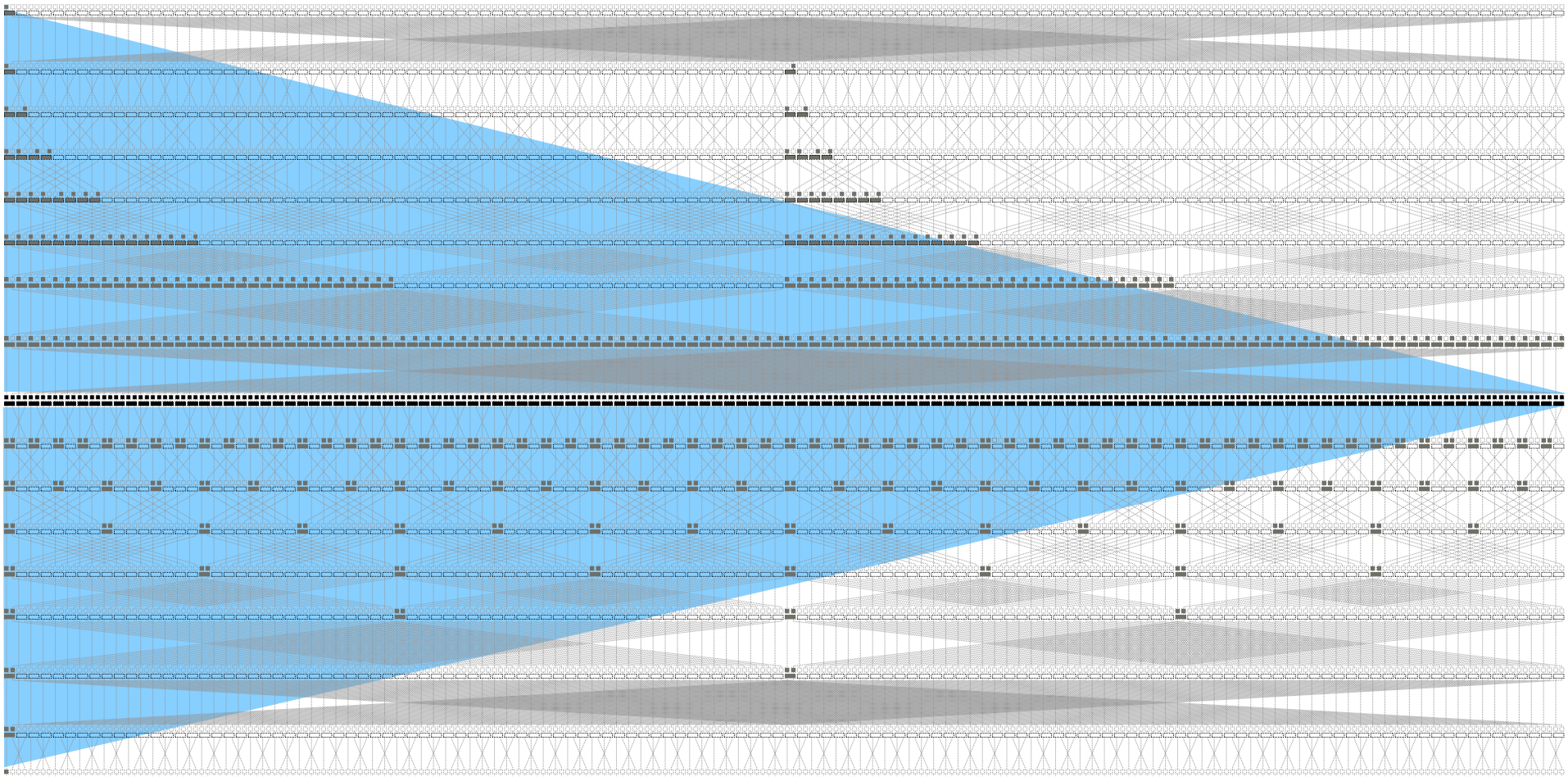
- Try all possible values, keep only the ones that satisfy the desired pattern
- Compute the cross-product of the highlighted sets and check if the differences satisfy the filtering conditions.
- Expand the sets by computing the cross-product of the highlighted sets.
- Compute the cross-product of the last two sets and check whether remaining 10 filtering conditions are satisfied or not.

Grouping - De-grouping



- Combine all possible values, keep only the ones that satisfy the desired pattern
- Compute the cross-product of the shown sets and check if the differences satisfy the filtering conditions.
- Expand the sets by computing the cross-product the highlighted sets.
- Compute the cross-product of last two sets and check whether remaining 10 filtering conditions satisfy or not.
- Calculate the differences through grouping and de-grouping part.

Attack on JH $d = 8$

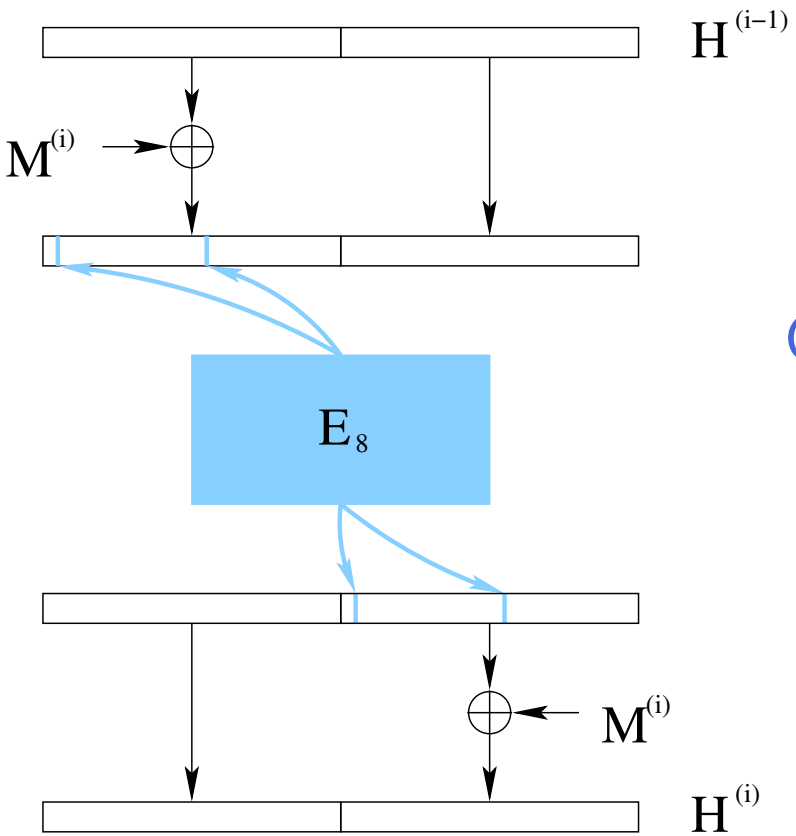


Attack on JH $d = 8$

Table: The overview of the attack on Round Function

Step	Size (bits)	Sets	Filtering Conditions	Pairs Remain	Time Complexity	Direction
0	8	128	1	$2^{11.75}$	$2^{15.84}$	*fwd*
1	16	64	1	$2^{19.59}$	$2^{23.50}$	*bck*
2	32	32	4	$2^{23.54}$	$2^{39.18}$	*fwd*
3	64	16	4	$2^{31.44}$	$2^{47.08}$	*fwd*
4	128	8	4	$2^{47.24}$	$2^{62.88}$	*fwd*
5	256	4	8	$2^{63.20}$	$2^{94.48}$	*fwd*
6	512	2	8	$2^{95.12}$	$2^{124.40}$	*fwd*
7	1024	1	46	$2^{10.38}$	$2^{190.24}$	*fwd* + *bck*

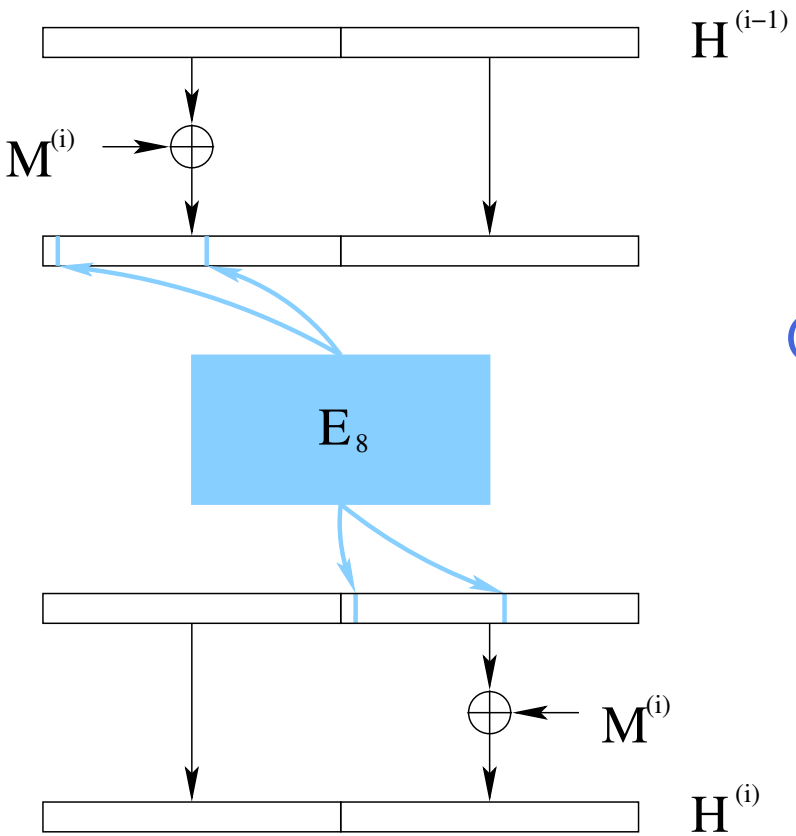
Attack on JH $d = 8$



Grouping and degrouping

- $2^{10.38} \cdot 2^{-1} \cdot (3/15)^2$
-

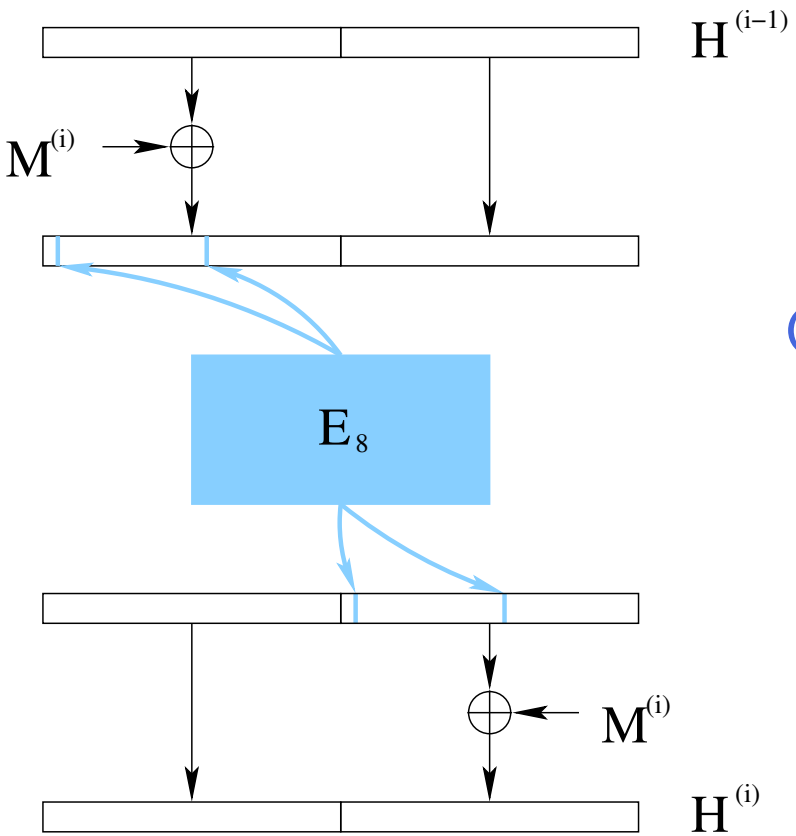
Attack on JH $d = 8$



Grouping and degrouping

- $2^{10.38} \cdot 2^{-1} \cdot (2^{-2.32})^2 \simeq 2^{4.74}$
-

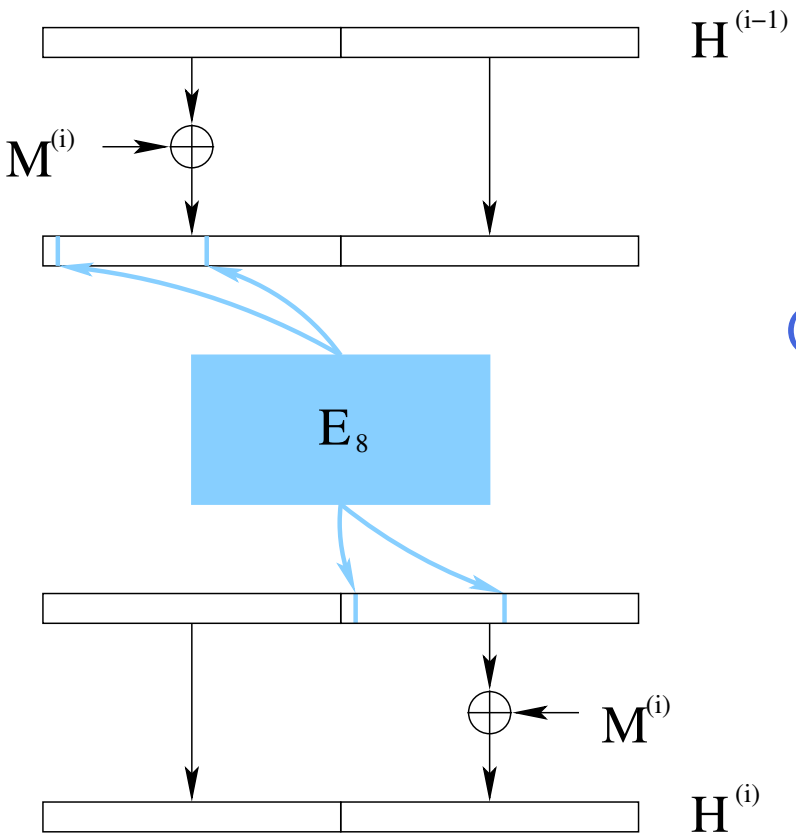
Attack on JH $d = 8$



Grouping and degrouping

- $2^{10.38} \cdot 2^{-1} \cdot (2^{-2.32})^2 \simeq 2^{4.74}$
- $2^{4.74} \cdot (1/3)$

Attack on JH $d = 8$

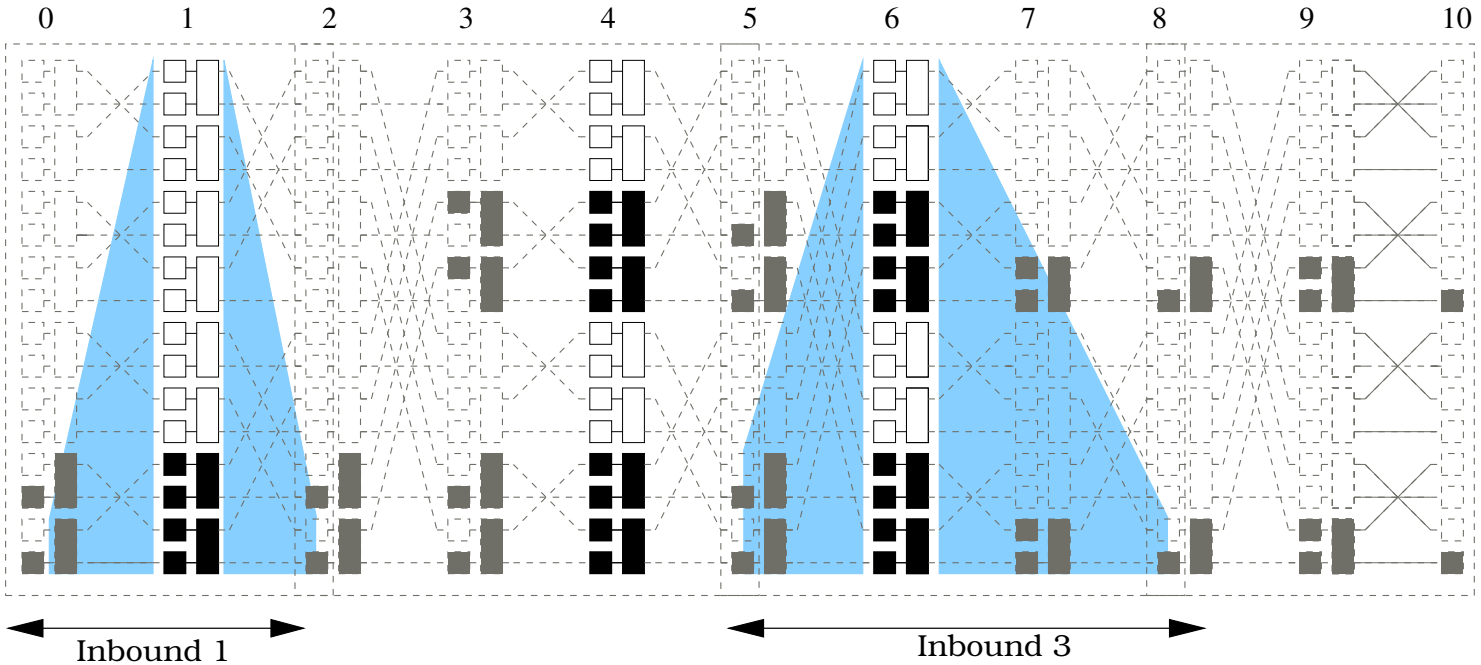


Grouping and degrouping

- $2^{10.38} \cdot 2^{-1} \cdot (2^{-2.32})^2 \simeq 2^{4.74}$
- $2^{4.74} \cdot (2^{-1.59}) \simeq 2^{3.15}$

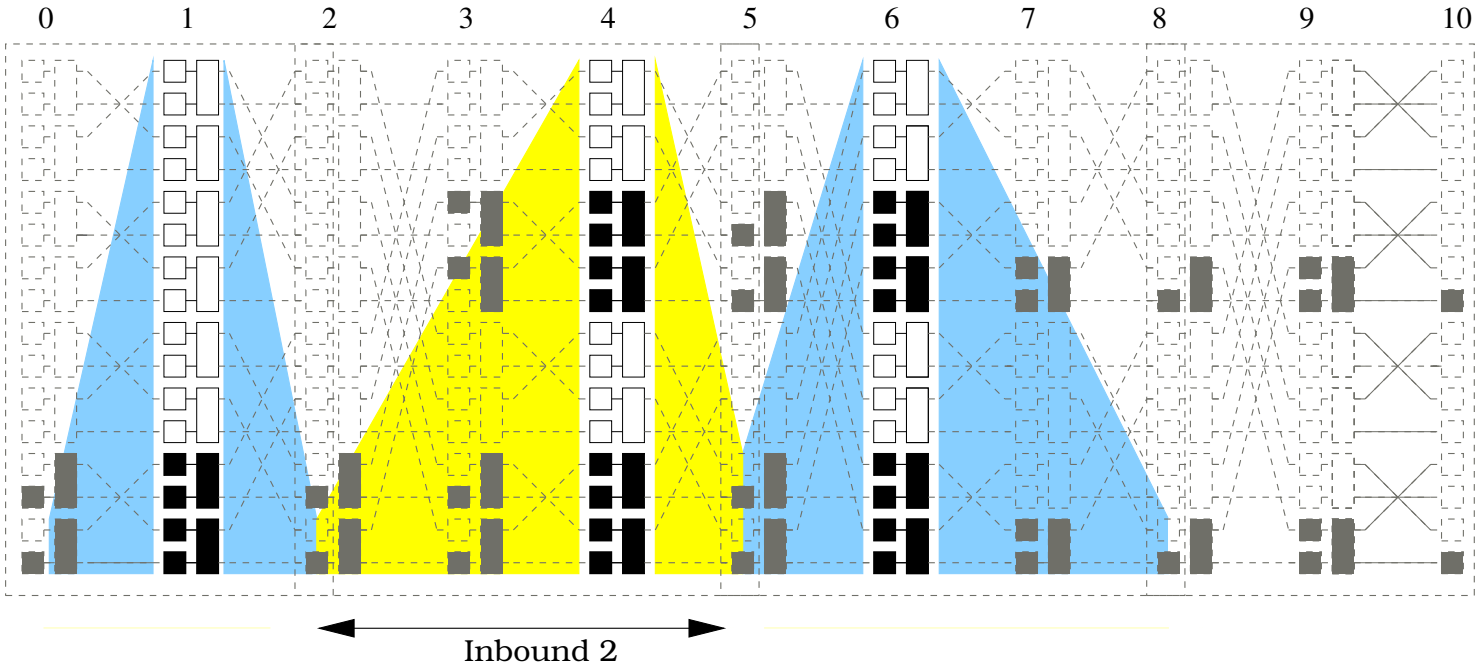
Outline

Overview of the Attack



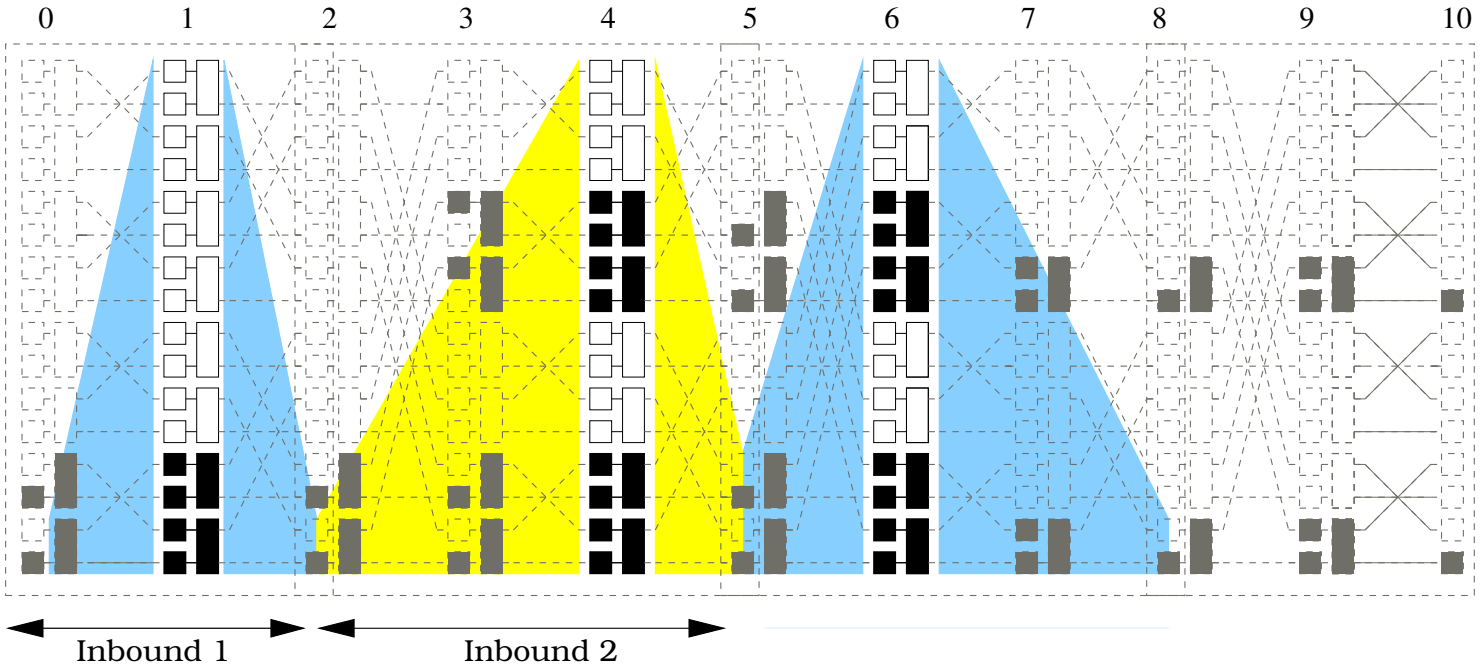
- Compute the first and the third inbound phases

Overview of the Attack



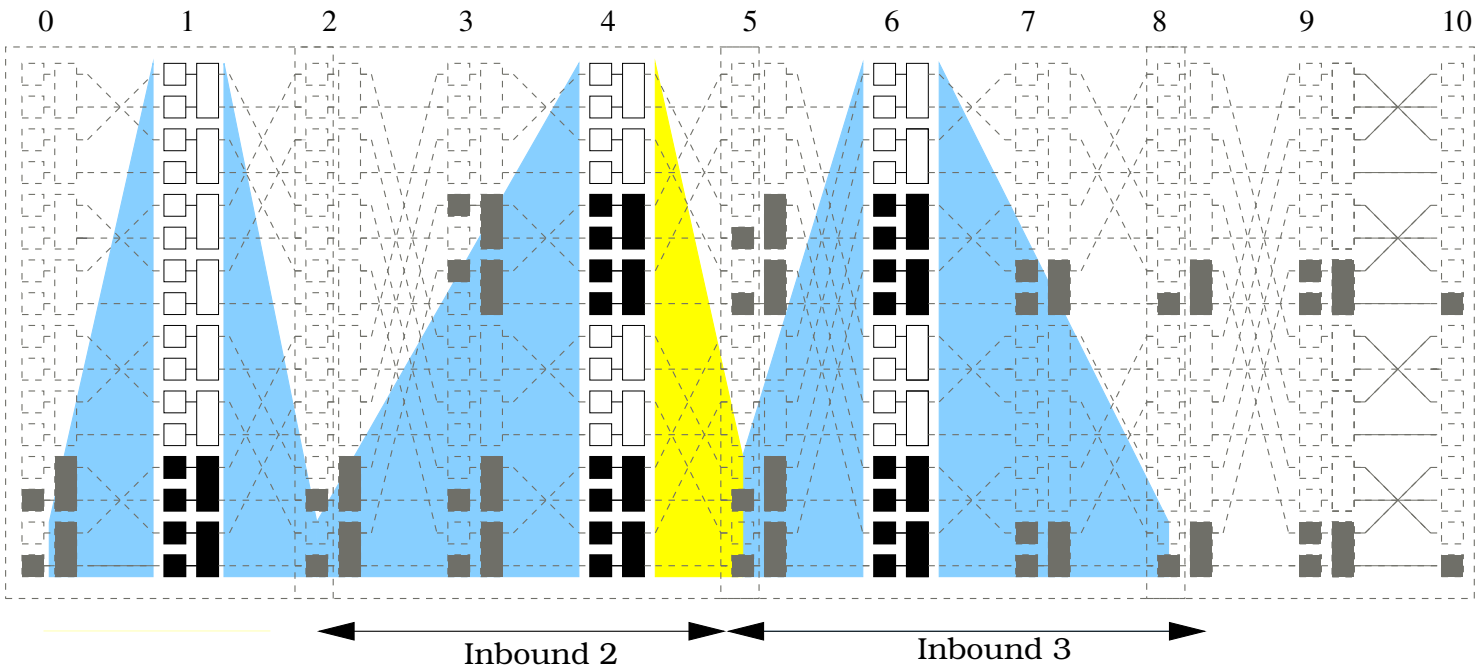
- Compute the first and the third inbound phases
- Compute the second inbound phase

Overview of the Attack



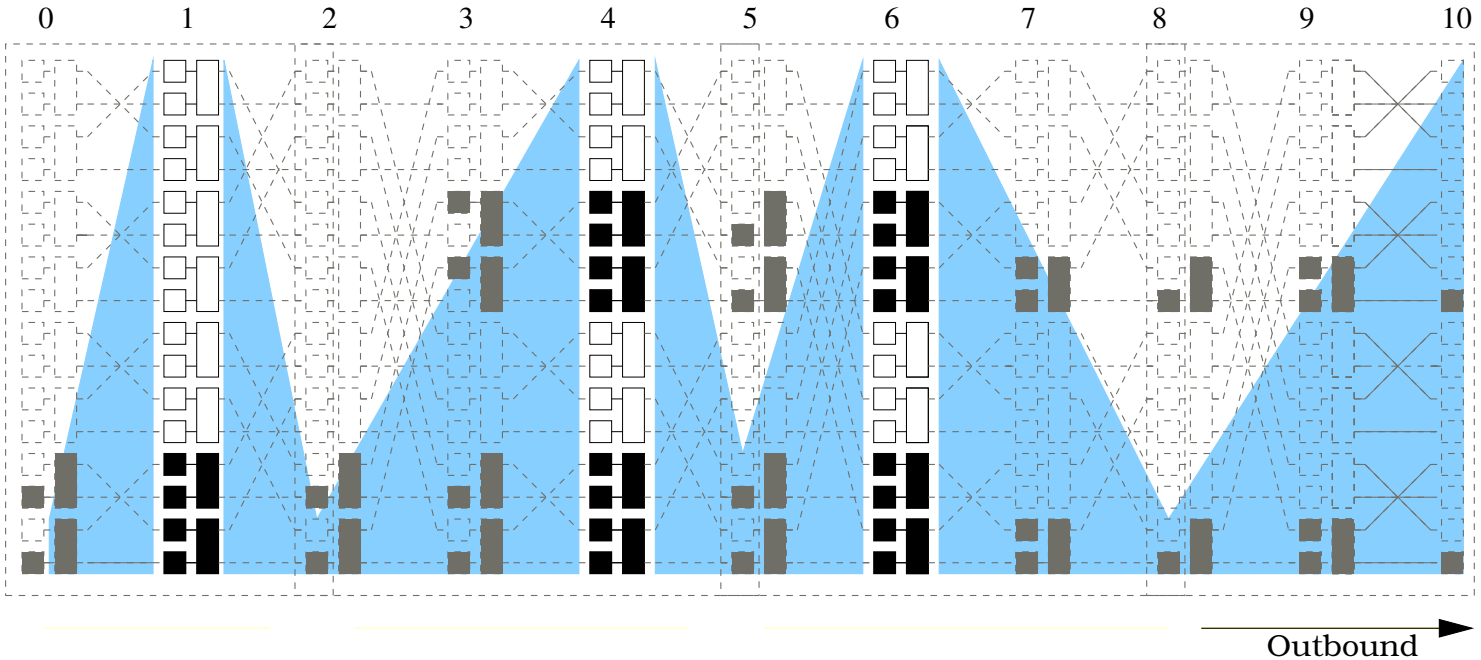
- Compute the first and the third inbound phases
- Compute the second inbound phase
- Merge the first and the second inbound phases

Overview of the Attack



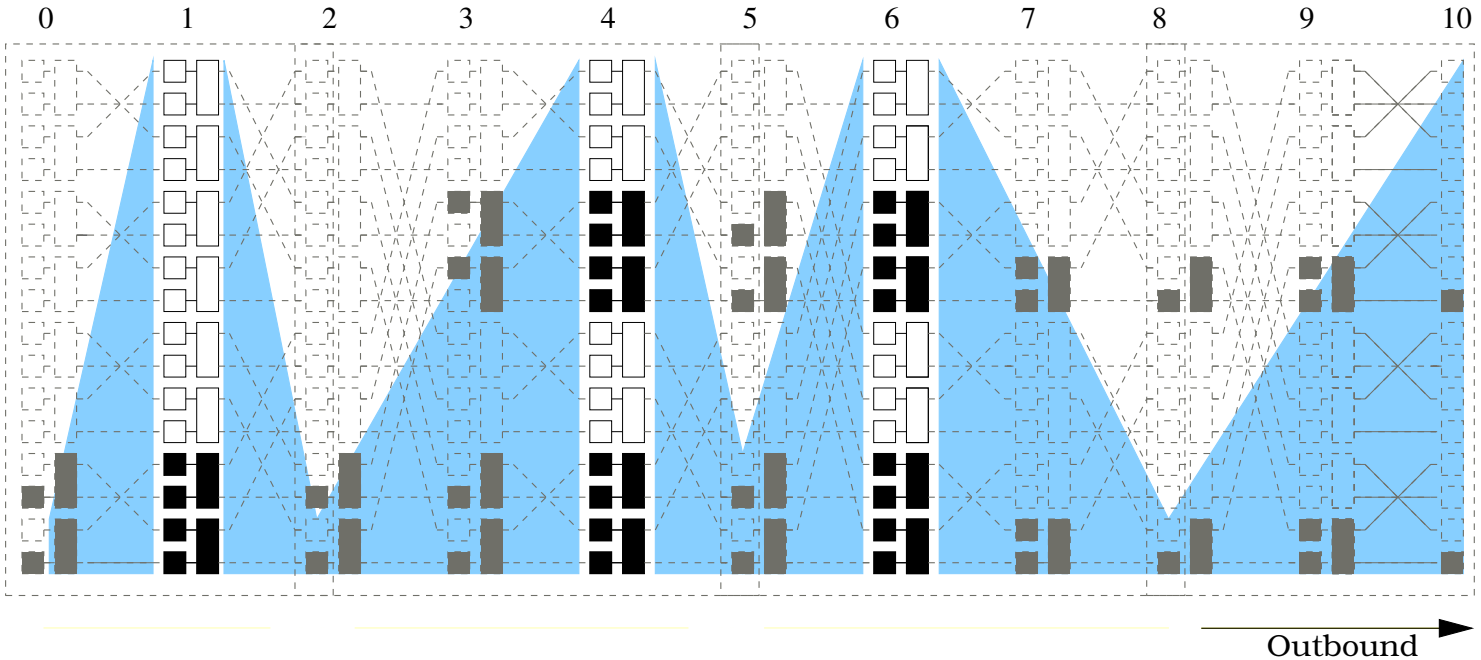
- Compute the first and the third inbound phases
- Compute the second inbound phase
- Merge the first and the second inbound phases
- Merge the second and the third inbound phases

Overview of the Attack



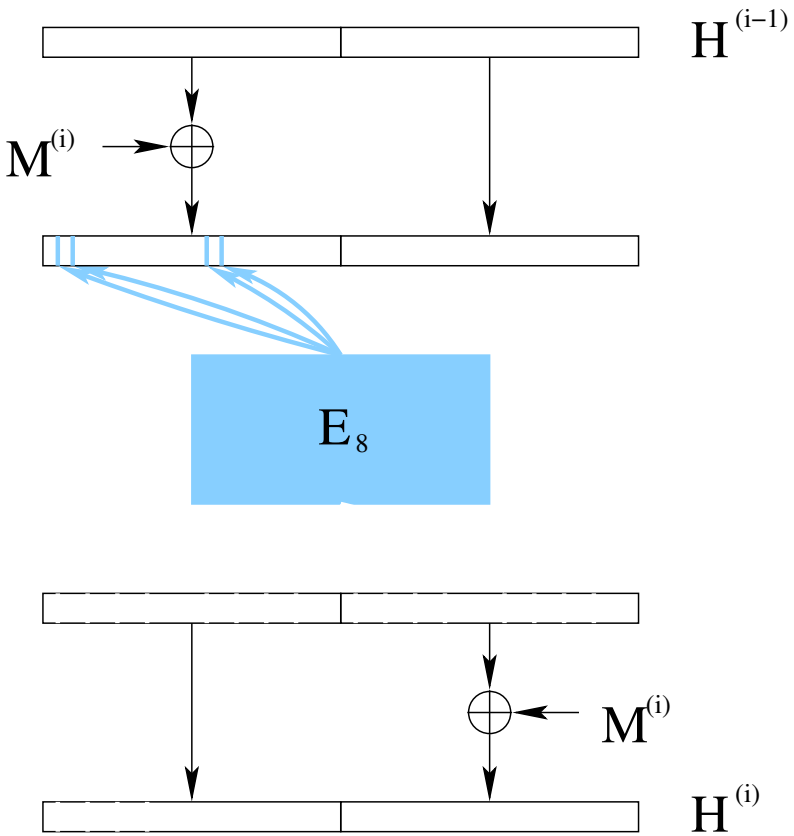
- Compute the first and the third inbound phases
- Compute the second inbound phase
- Merge the first and the second inbound phases
- Merge the second and the third inbound phases
- Compute the outbound phase

Overview of the Attack



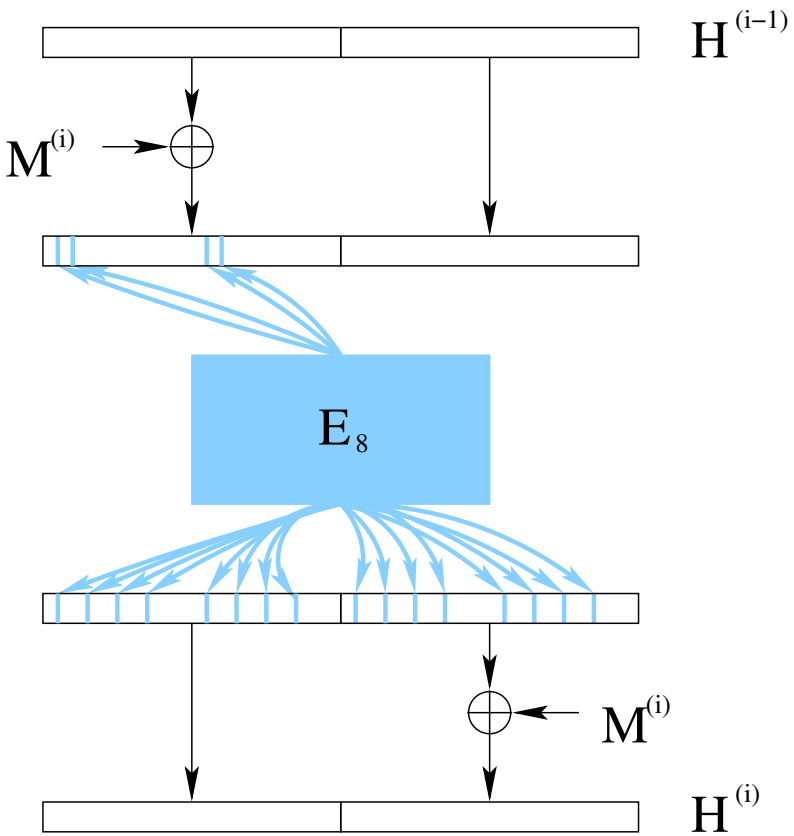
- Compute the first and the third inbound phases
- Compute the second inbound phase
- Merge the first and the second inbound phases
- Merge the second and the third inbound phases
- Compute the outbound phase
- **Result:** Reduced round semi-free-start near-collisions on the compression function of *JH*

Attack on JH $d = 8$



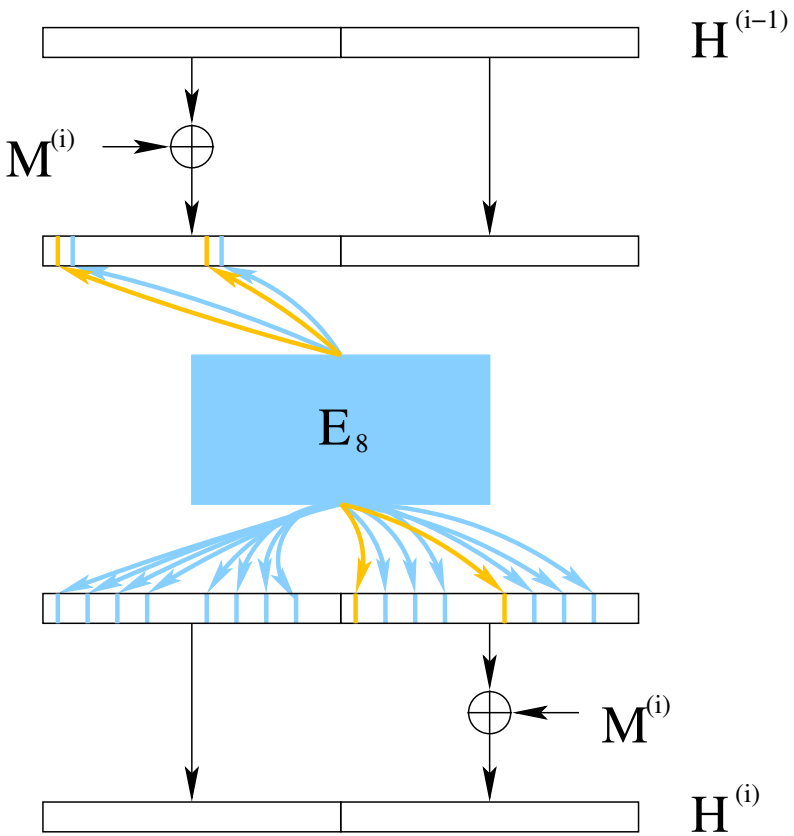
- 4 bit message difference

Attack on JH $d = 8$



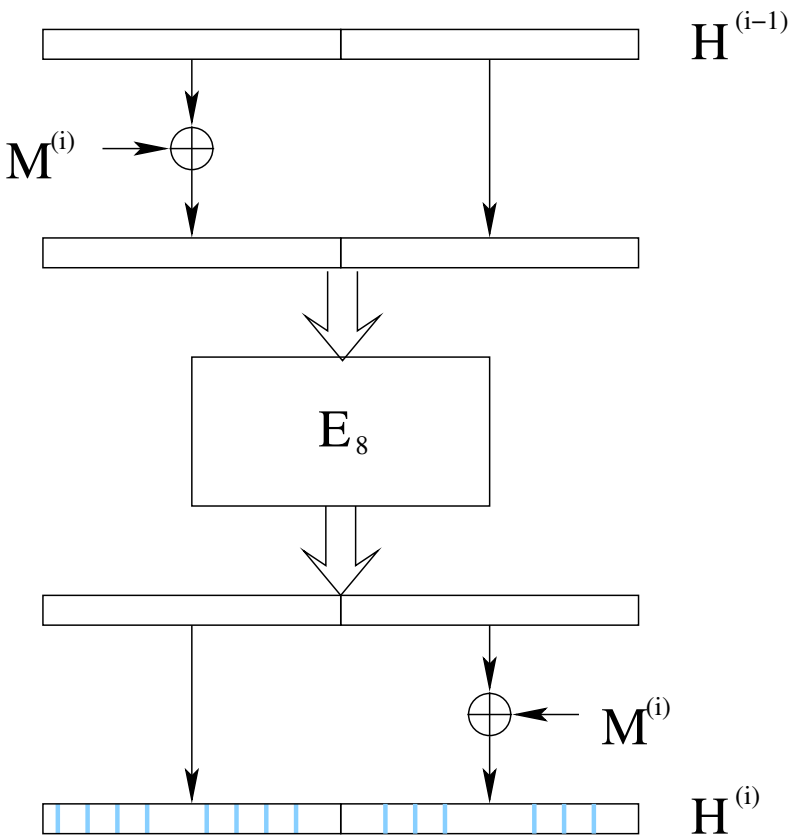
- 4 bit message difference
- 16 bit difference from output

Attack on JH $d = 8$



- 4 bit message difference
- 16 bit difference from output
- Two of which collide

Attack on JH $d = 8$



- 4 bit message difference
- 16 bit difference from output
- Two of which collide
- $4 + 16 - 4 = 16$ bit difference left in the state.

Outline

Results and Conclusion

Table: Summary of results for JH (CFC - Compression Function Call)

	#Rounds	Time Complexity	Memory Complexity	Collision Type
Hash Func.	16/35.5	$2^{178.24}$ CFC	$2^{101.12}$ byte	semi-free-start collision
Comp. Func.	19/35.5	$2^{156.77}$ CFC	$2^{143.70}$ byte	semi-free-start near-collision (1008 bits)
Comp. Func.	22/35.5	$2^{156.56}$ CFC	$2^{143.70}$ byte	semi-free-start near-collision (768 bits)

Results and Conclusion

Table: Summary of results for JH (CFC - Compression Function Call)

	#Rounds	Time Complexity	Memory Complexity	Collision Type
Hash Func.	16/35.5	$2^{178.24}$ CFC	$2^{101.12}$ byte	semi-free-start collision
Comp. Func.	19/35.5	$2^{156.77}$ CFC	$2^{143.70}$ byte	semi-free-start near-collision (1008 bits)
Comp. Func.	22/35.5	$2^{156.56}$ CFC	$2^{143.70}$ byte	semi-free-start near-collision (768 bits)

Further Work

Working on ways to improve the complexity to increase the attack with a couple of rounds

Results and Conclusion

Thank You!...