Linear Propagation in Efficient Guess-and-Determine Attacks

Maria Eichlseder[1] Florian Mendel[1] Tomislav Nad[1]
Vincent Rijmen[2] Martin Schläffer[1]

[1]IAIK, Graz University of Technology, Austria

[2]ESAT/COSIC, KU Leuven and iMinds, Belgium

WCC 2013

Motivation

Cryptanalysis of hash functions

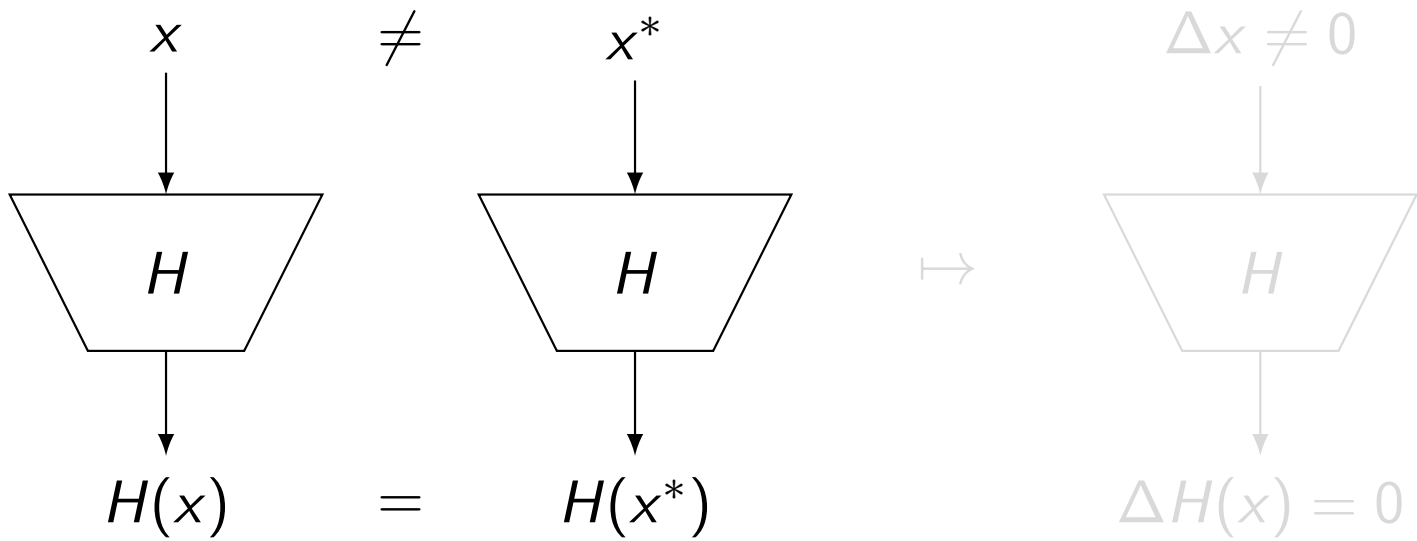
- attacks on MD4, MD5, SHA-1 done by hand [WY05]
- not feasible for SHA-2 or SHA-3
- need for automated analysis tools

Recent results based on automated tools:

- Leurent's analysis of BLAKE, Skein [Leu12]
- Dinur, Dunkelman and Shamir's analysis of SHA-3 [DDS12]
- our tool, previously used for SHA-1 [DR06], SHA-2 [MNS11], RIPEMD-128/160 [MNS12, MNSS12]

Differential collision attacks

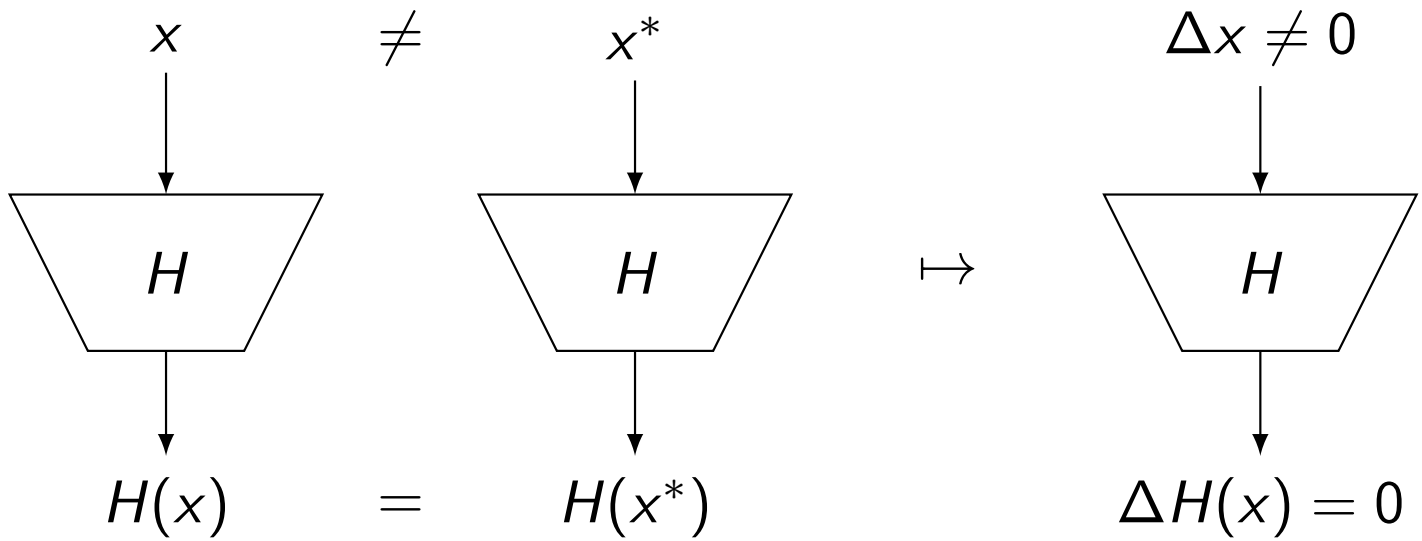
- Applications require collision resistant hash functions



- Evaluate resistance against differential cryptanalysis
  "What happens to Δx inside the hash function?"

Differential collision attacks

- Applications require collision resistant hash functions



- Evaluate resistance against differential cryptanalysis
  "What happens to Δx inside the hash function?"

Describing differences

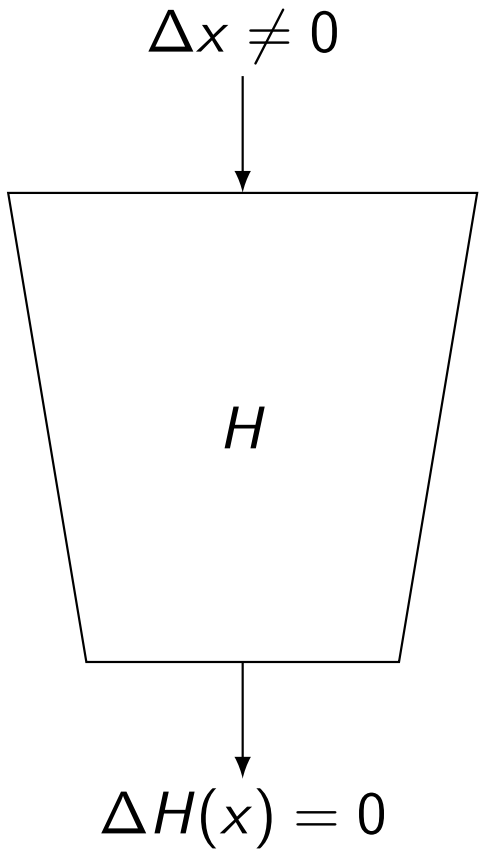
Differential characteristic describes differences and conditions for messages, hash and intermediate variables

- Classical differences: xor, signed, modular, ...
- Finer description: generalized conditions [DR06]

(x, x^*)	$(0,0)$	$(0,1)$	$(1,0)$	$(1,1)$
-	✓	·	·	✓
x	·	✓	✓	·
0	✓	·	·	·
1	·	·	·	✓
?	✓	✓	✓	✓
#	·	·	·	·
⋮				

16 bitwise conditions

Outline of the attack

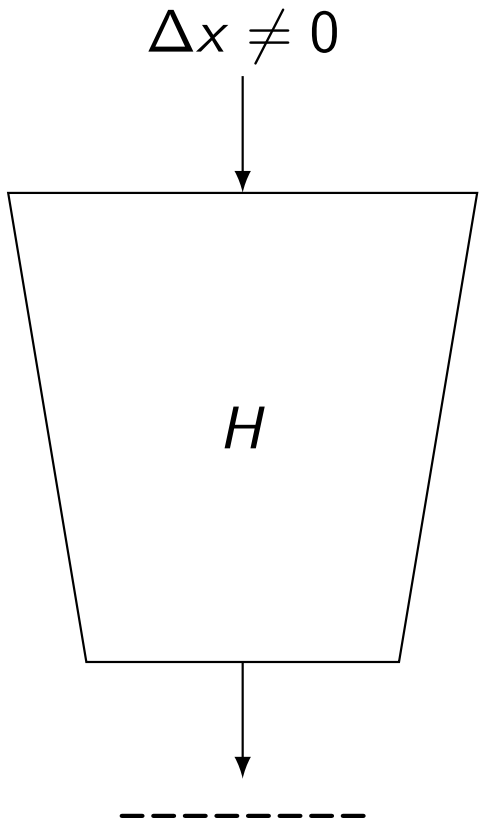


1. starting difference

2. differential characteristic
   - by hand
   - low weight code words
   - guess and determine

3. message pair
   - random message pair
   - guess and determine

Outline of the attack

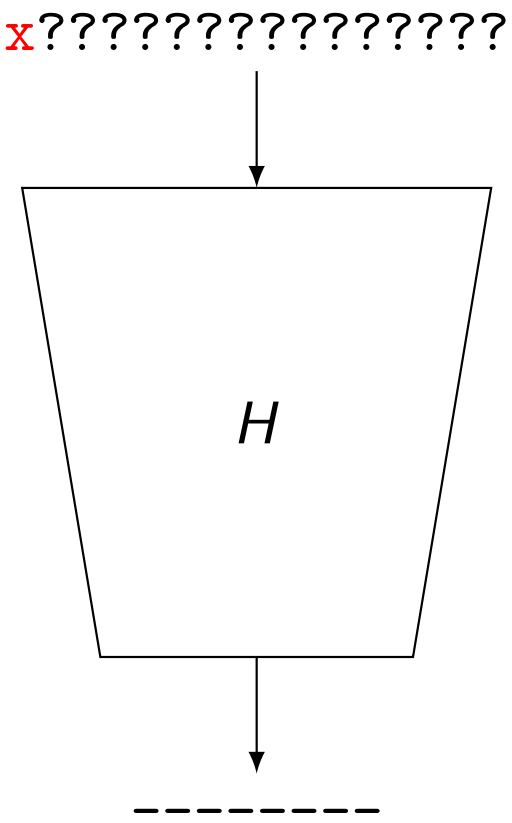


1. starting difference

2. differential characteristic
   - by hand
   - low weight code words
   - guess and determine

3. message pair
   - random message pair
   - guess and determine

Outline of the attack

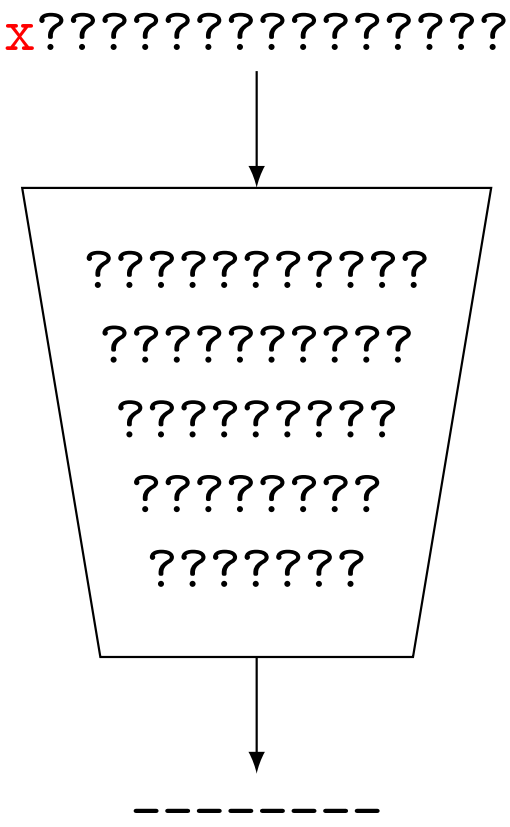


1. starting difference

2. differential characteristic
   - by hand
   - low weight code words
   - guess and determine

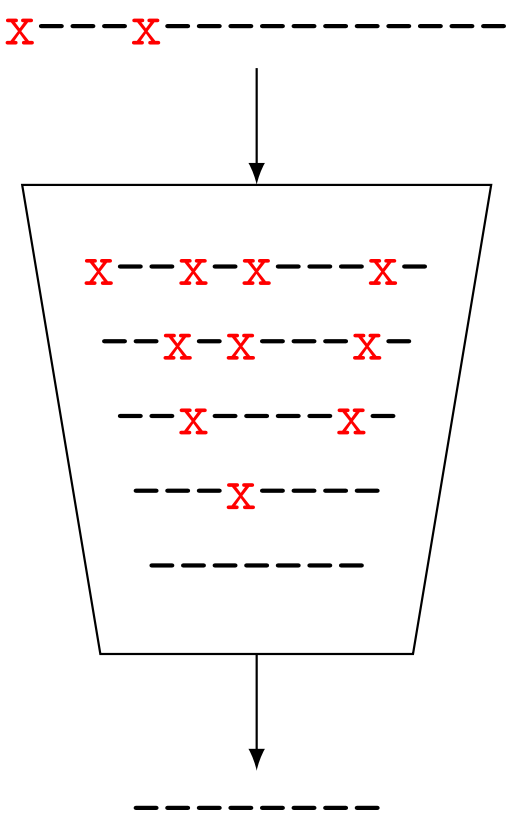
3. message pair
   - random message pair
   - guess and determine

Outline of the attack



1. starting difference
2. differential characteristic
   - by hand
   - low weight code words
   - guess and determine
3. message pair
   - random message pair
   - guess and determine

Outline of the attack

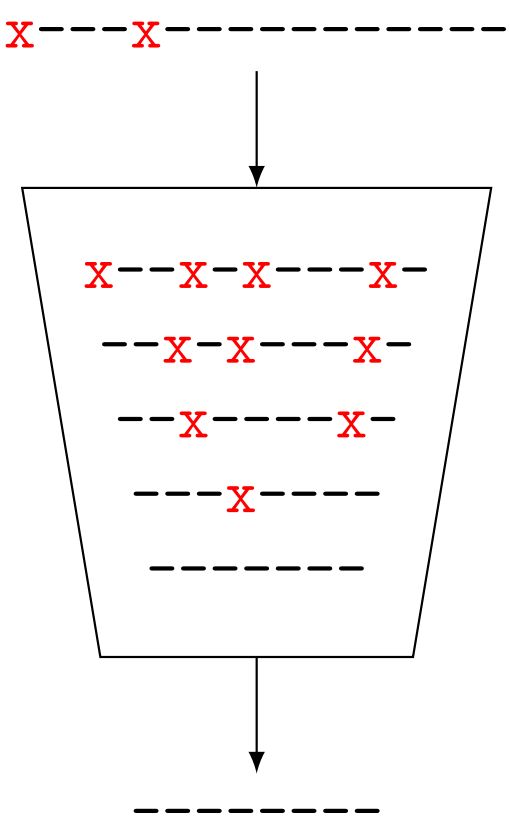


1. starting difference

2. differential characteristic
   - by hand
   - low weight code words
   - guess and determine

3. message pair
   - random message pair
   - guess and determine

Outline of the attack

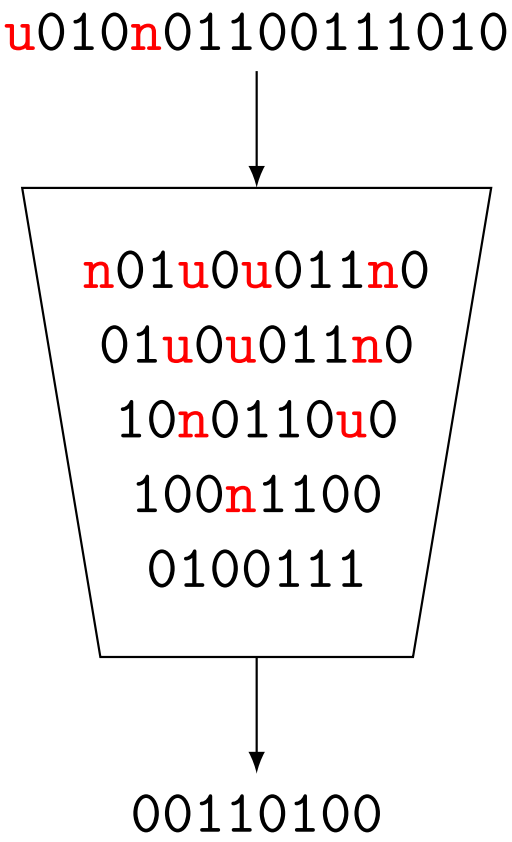


1. starting difference

2. differential characteristic
   - by hand
   - low weight code words
   - guess and determine

3. message pair
   - random message pair
   - guess and determine

Outline of the attack



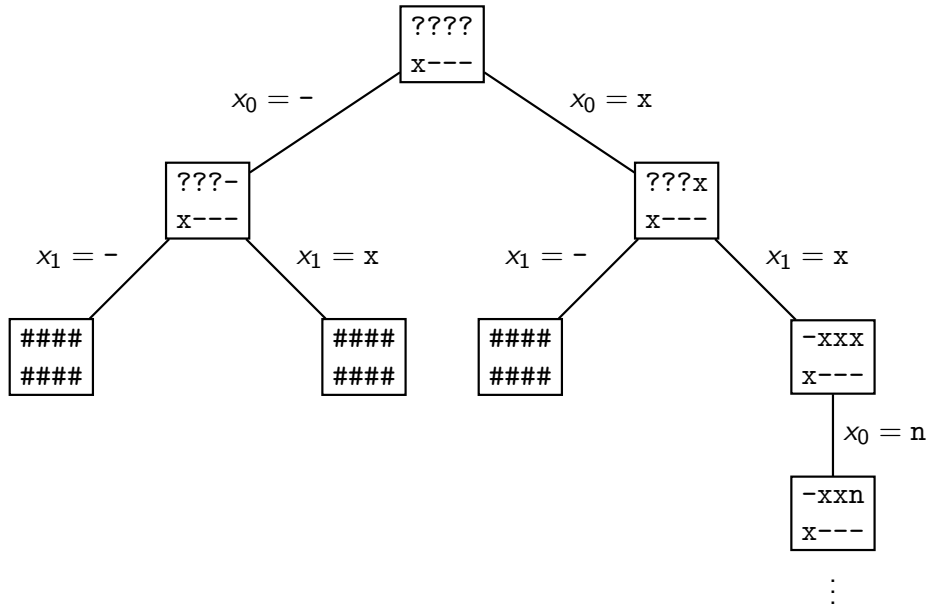
1. starting difference

2. differential characteristic
   - by hand
   - low weight code words
   - guess and determine

3. message pair
   - random message pair
   - guess and determine

Guess and determine

Depth first search in the guessing tree:

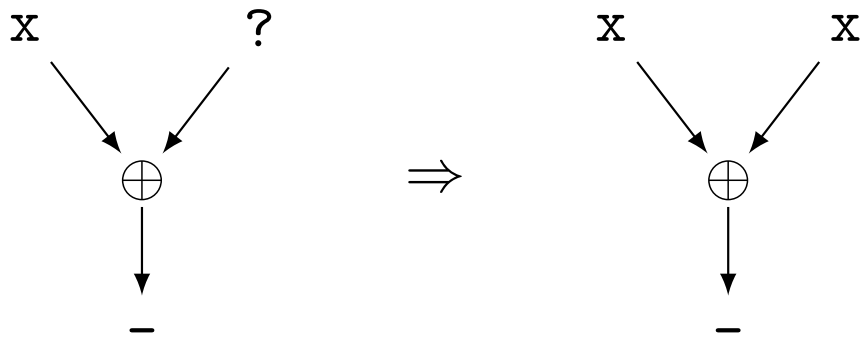


Improved by

- propagating consequences of each guess
- guessing strategy
- backtracking strategy

Propagation of conditions

Difference conditions imply additional conditions for consistency:

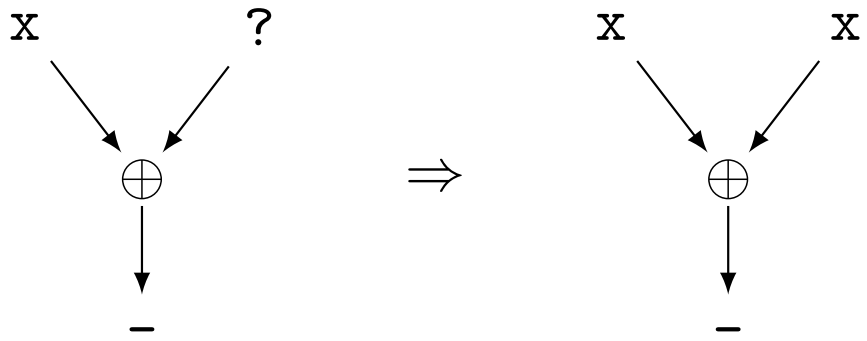


- Perfect propagation → infeasible
- Bitsliced propagation → tiny substeps, only locally perfect

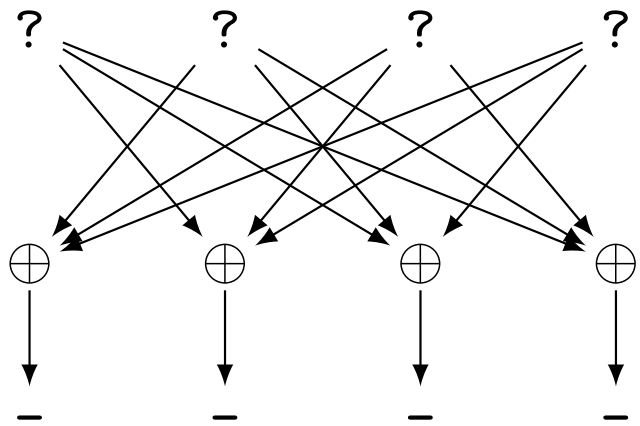
- Linear propagation → much larger substeps, more global

Propagation of conditions

Difference conditions imply additional conditions for consistency:



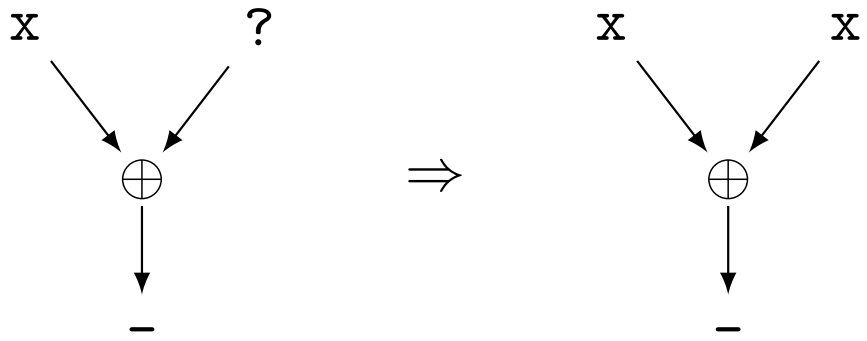
- Perfect propagation $\rightarrow$ infeasible
- Bitsliced propagation $\rightarrow$ tiny substeps, only locally perfect



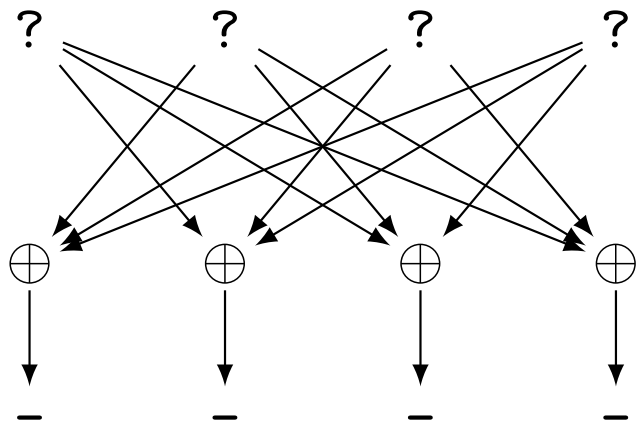
- Linear propagation $\rightarrow$ much larger substeps, more global

Propagation of conditions

Difference conditions imply additional conditions for consistency:



- Perfect propagation $\rightarrow$ infeasible
- Bitsliced propagation $\rightarrow$ tiny substeps, only locally perfect



- Linear propagation $\rightarrow$ much larger substeps, more global

Linear propagation

- Simultaneous equations for
  - affine linear function definition $\begin{pmatrix} L & 0 \\ 0 & L \end{pmatrix} \cdot \begin{pmatrix} z \\ z^* \end{pmatrix} = \begin{pmatrix} d \\ d \end{pmatrix}$
  - affine linear generalized conditions $\begin{pmatrix} C & C^* \end{pmatrix} \cdot \begin{pmatrix} z \\ z^* \end{pmatrix} = \begin{pmatrix} b \end{pmatrix}$
- Gauss-Jordan elimination to get reduced row echelon form

$$\begin{pmatrix} \diagdown & \\ & \diagup \end{pmatrix} \cdot \begin{pmatrix} z \\ z^* \end{pmatrix} = \begin{pmatrix} \square \end{pmatrix}$$

- Translate back to generalized conditions

Linear propagation

- Simultaneous equations for
  - affine linear function definition $\begin{pmatrix} L & 0 \\ 0 & L \end{pmatrix} \cdot \begin{pmatrix} z \\ z^* \end{pmatrix} = \begin{pmatrix} d \\ d \end{pmatrix}$
  - affine linear generalized conditions $\begin{pmatrix} C & C^* \end{pmatrix} \cdot \begin{pmatrix} z \\ z^* \end{pmatrix} = \begin{pmatrix} b \end{pmatrix}$
- Gauss-Jordan elimination to get reduced row echelon form

$$\begin{pmatrix} \blacktriangle \end{pmatrix} \cdot \begin{pmatrix} z \\ z^* \end{pmatrix} = \begin{pmatrix} \square \end{pmatrix}$$

- Translate back to generalized conditions

Linear propagation in practice

- Nonlinear hash function parts
  - cover with bitsliced propagation
  - loosely coupled equation systems
- Nonlinear conditions
  - supplement with bitsliced propagation, or
  - ignore, since they are rare
- Guessing strategy
  - prefer bits with many simple equations
- Matrix implementation
  - step function size
  - incremental changes $\rightarrow$ incremental Gauss-Jordan

Evaluation

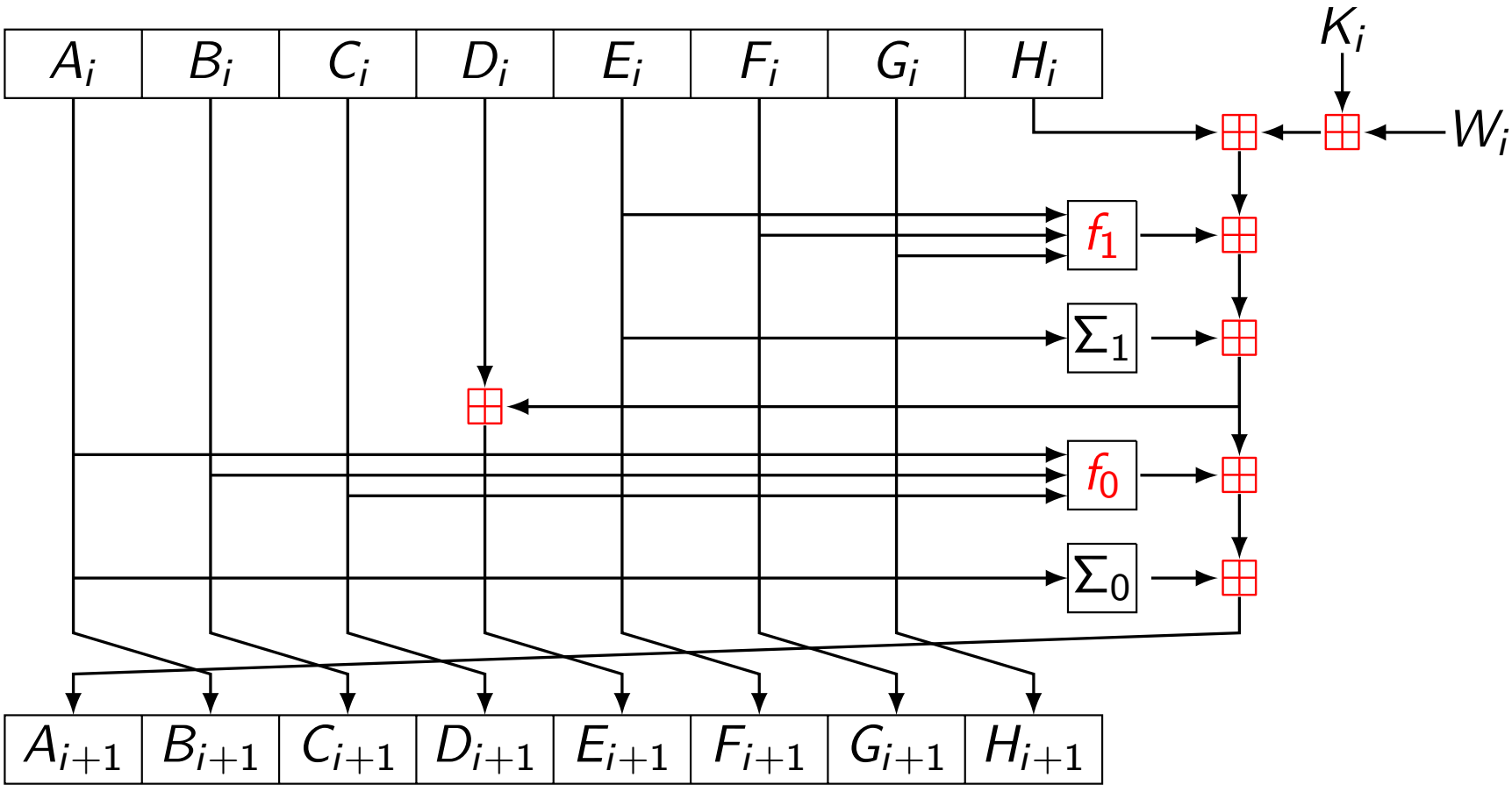
Quality of propagation method:

- Reduction of solution space
- For one sample: $I_M = \log_2 \frac{\text{solutions before propagation}}{\text{solutions after propagation}}$
- If $I_{\text{diff}} = I_{\text{bitslice}} - I_{\text{linear}} < 0 \Rightarrow$ new linear method better
- Plot cumulative distribution function

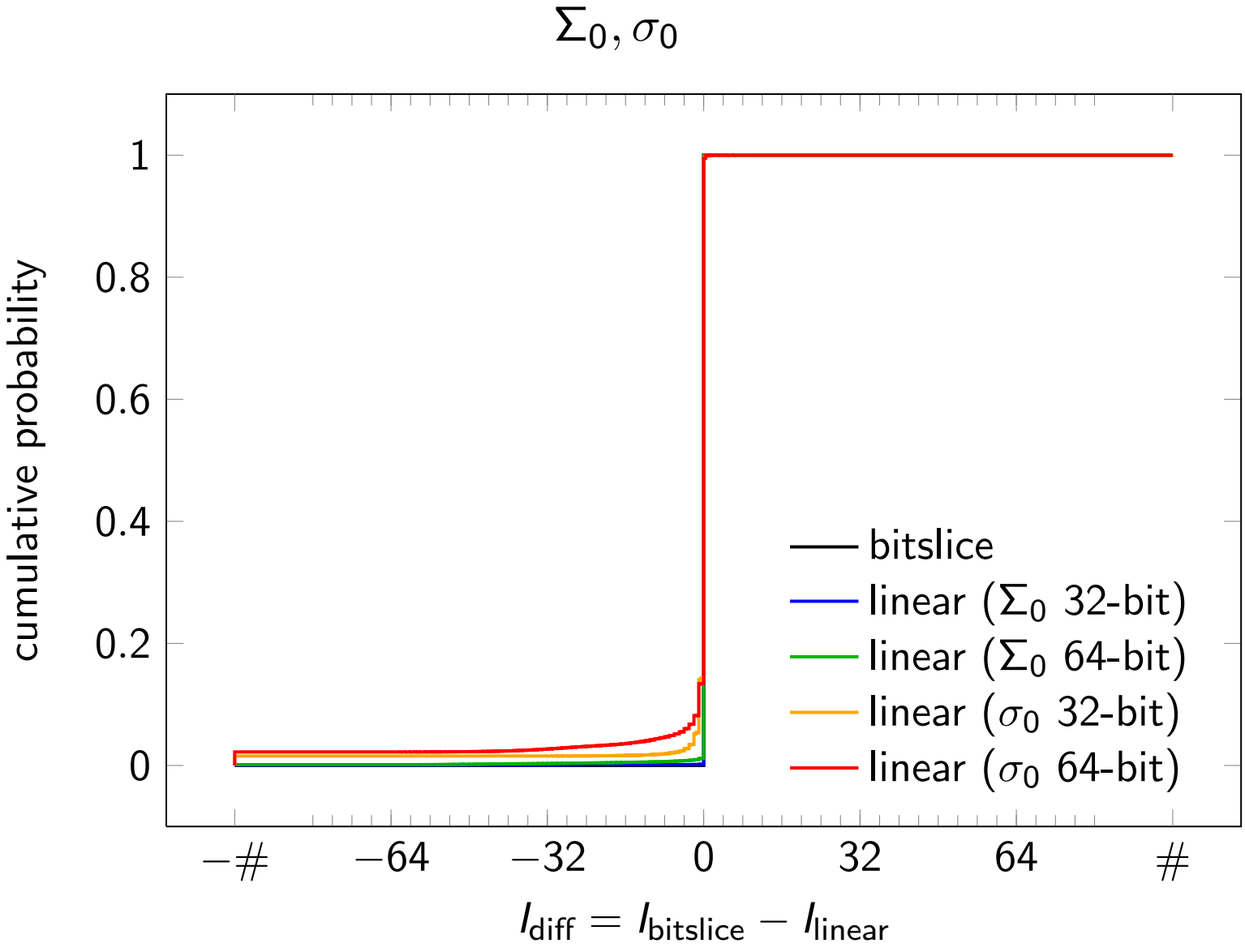
Efficiency of propagation method:

- Runtime per propagation
- Overall search runtime

Evaluation for Σ_i, σ_i functions of SHA-2

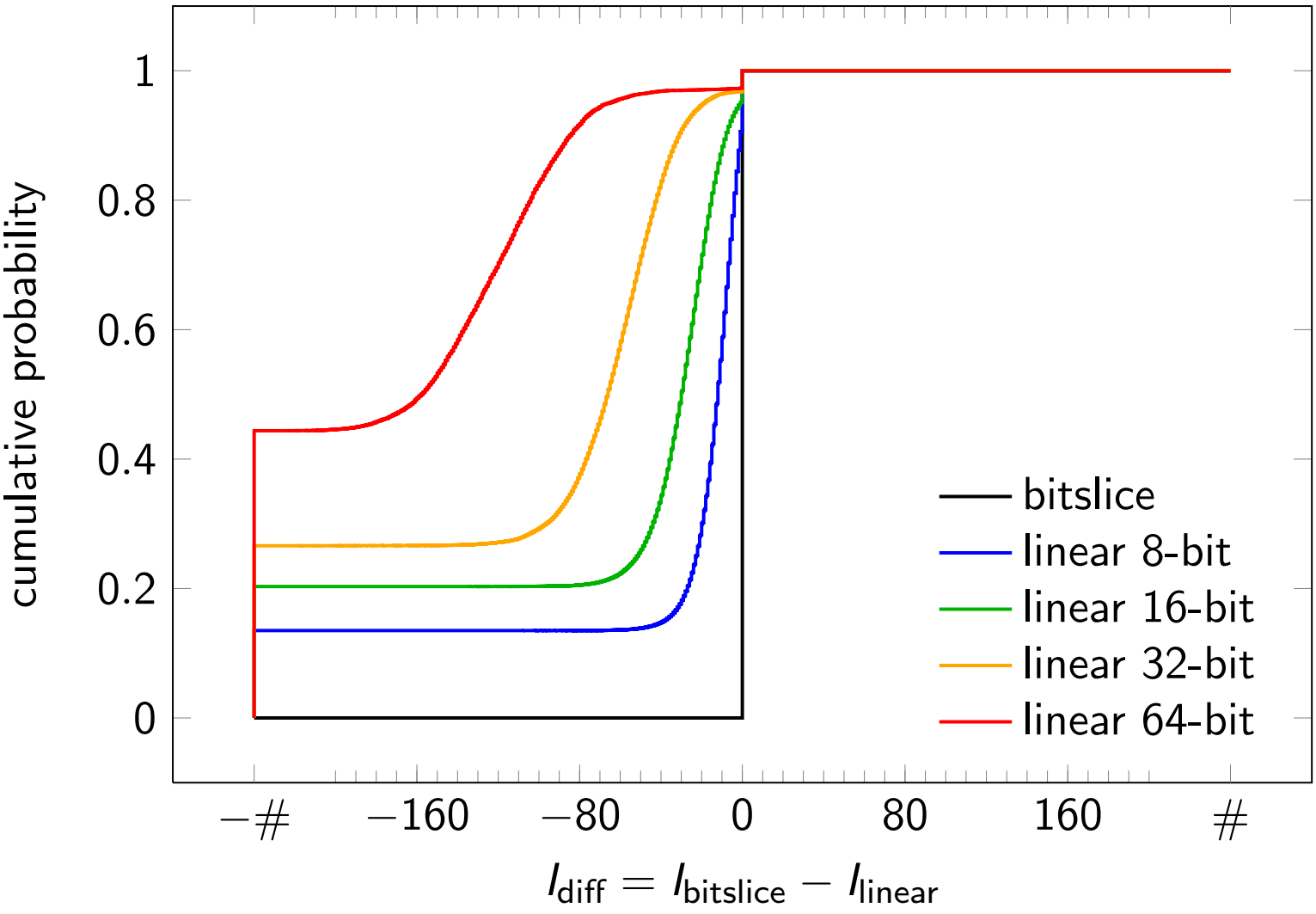


Evaluation for Σ_i, σ_i functions of SHA-2



Evaluation for 2 rounds of SHA-3/Keccak

$\iota \circ \chi \circ \pi \circ \rho \circ \theta$



Evaluation for 2 rounds of SHA-3/Keccak

		2h		16h	
lane size		8-bit	16-bit	32-bit	64-bit
iterations/sec	bitslice	21158	21542	14612	9566
	linear	4272	2102	769	222
solutions found	bitslice	333	4	0	0
	linear	1069	41	4	0

Combined with 2-round kernel path $\rightarrow$ 4-round collision

Conclusion

- Propagation is essential for guess-and-determine attacks
- New linear method provides globally better results
- Tradeoff between quality and runtime
- Much better than bitsliced method for SHA-3

Future

- Integration of nonlinear parts
- Multiple rounds

Linear Propagation in Efficient Guess-and-Determine Attacks

Maria Eichlseder[1] Florian Mendel[1] Tomislav Nad[1]
Vincent Rijmen[2] Martin Schläffer[1]

[1]IAIK, Graz University of Technology, Austria

[2]ESAT/COSIC, KU Leuven and iMinds, Belgium

WCC 2013

Bibliography I


Itai Dinur, Orr Dunkelman, and Adi Shamir.
Self-differential cryptanalysis of up to 5 rounds of SHA-3.
IACR Cryptology ePrint Archive, 2012:672, 2012.


Christophe De Cannière and Christian Rechberger.
Finding SHA-1 characteristics: General results and applications.
In *ASIACRYPT*, pages 1–20, 2006.


Gaëtan Leurent.
Construction of differential characteristics in ARX designs – application to Skein.
IACR Cryptology ePrint Archive, 2012:668, 2012.

Bibliography II


Florian Mendel, Tomislav Nad, and Martin Schläffer.
Finding SHA-2 characteristics: Searching through a minefield of contradictions.
In *ASIACRYPT*, pages 288–307, 2011.


Florian Mendel, Tomislav Nad, and Martin Schläffer.
Collision attacks on the reduced dual-stream hash function RIPEMD-128.
In *FSE*, pages 226–243, 2012.


Florian Mendel, Tomislav Nad, Stefan Scherz, and Martin Schläffer.
Differential attacks on reduced RIPEMD-160.
In *ISC*, pages 23–38, 2012.

Bibliography III


Xiaoyun Wang and Hongbo Yu.
How to break MD5 and other hash functions.
In *EUROCRYPT*, pages 19–35, 2005.