Boomerang Distinguisher for the SIMD-512 Compression Function

Florian Mendel and Tomislav Nad

Institute for Applied Information Processing and Communications (IAIK)
Graz University of Technology
Inffeldgasse 16a, A-8010 Graz, Austria

Tomislav.Nad@iaik.tugraz.at

Outline

Outline

SHA-3 Competition

- Organized by NIST [Nat07]
- Successor for SHA-1 and SHA-2
- 64 submissions
- 51 round 1 candidates
- 14 round 2 candidates
- 5 finalists

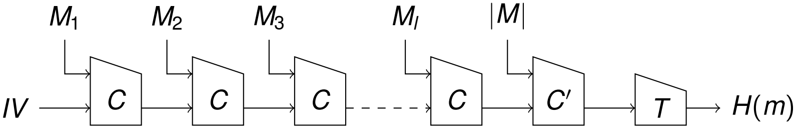
Outline

SIMD Is a Message Digest[LBF08]

- Designed by Gaëtan Leurent, Charles Bouillaguet and Pierre-Alain Fouque
- Round 2 candidate
- Message block
  - SIMD-256: 512 bits
  - SIMD-512: 1024 bits
- Inner state (wide-pipe)
  - SIMD-256: 16 32-bit words
  - SIMD-512: 32 32-bit words

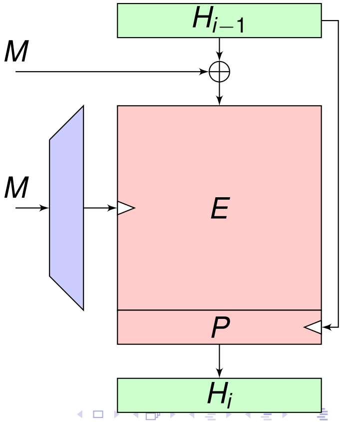
The SIMD Hash Function

- Similar to Chop-MD
- Internal state is twice as large as the output
- Output transformation: truncation T



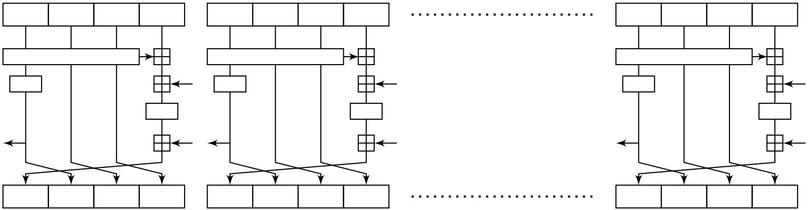
The SIMD Compression Function (1/2)

- Modified Davis-Meyer construction
- Expanded message size: $8 \cdot$ blocksize
- Strong security in the message expansion

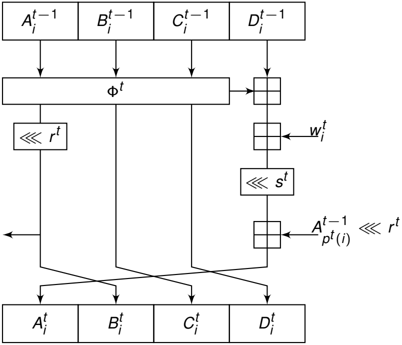


The SIMD Compression Function (2/2)

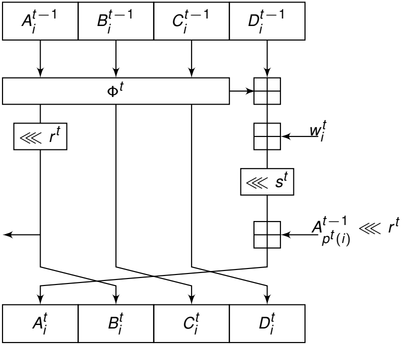
- Based on a Feistel structure; similar to MD5
- SIMD-256: 4 times the step function in parallel
- SIMD-512: 8 times the step function in parallel
- 32 steps plus 4 steps in the feed-forward



Update Function at Step t



Update Function at Step t



$$A_i^t = (D_i^{t-1} \boxplus w_i^t \boxplus \Phi(A_i^{t-1}, B_i^{t-1}, C_i^{t-1})) \lll s^t \boxplus (A_{p^t(i)}^{t-1} \lll r^t)$$

$$B_i^t = A_i^{t-1} \lll r^t$$

$$C_i^t = B_i^{t-1}$$

$$D_i^t = C_i^{t-1}$$

Φ is either IF or MAJ

Results on SIMD-512

Distinguisher

- Mendel and Nad [MN09]
  - Full compression function (complexity: 2^{427}) $\rightarrow$ tweaked!
- Nikolić et al. [INS10]
  - 12 out of 32 steps (complexity: 2^{236})
- Yu and Wang [YW11]
  - Full compression function (complexity: 2^{398})

Free-start near-collision

- Yu and Wang [YW11]
  - 24 out of 32 steps (complexity: 2^{208})

Our Contribution

Application of Higher-Order Differentials to SIMD-512

- Non-random properties for the permutation of SIMD-512
- Extend technique to overcome the feed-forward of SIMD-512
- Non-random properties for the compression function of SIMD-512

Outline

Higher-Order Differentials

- Introduced by Lai in [Lai94]
- First applied to block ciphers by Knudsen [Knu94]
- Recently applied to SHA-2 [BLMN11] and several SHA-3 candidates
  - BLAKE [BNR11], Hamsi [BC10], Keccak [BC10], Luffa [WHYK10], ...

Higher-Order Differentials: Basic Definitions

Definition

Let $(S, +)$ and $(T, +)$ be abelian groups. For a function $f\colon S \to T$, the derivative at a point $a_1 \in S$ is defined as

$$\Delta_{(a_1)} f(y) = f(y + a_1) - f(y) .$$

The i-th derivative of f at $(a_1, a_2, \ldots, a_i)$ is then recursively defined as

$$\Delta_{(a_1,\ldots,a_i)} f(y) = \Delta_{(a_i)}(\Delta_{(a_1,\ldots,a_{i-1})} f(y)) .$$

Higher-Order Differentials: Basic Definitions

Definition

A differential of order i for a function $f\colon S \rightarrow T$ is an $(i+1)$-tuple $(a_1, a_2, \ldots, a_i; b)$ such that

$$\Delta_{(a_1,\ldots,a_i)} f(y) = b\,.$$

Higher-Order Differential Collision

When applying differential cryptanalysis to a hash function, a collision for the hash function corresponds to a pair of inputs with output difference zero.

Higher-Order Differential Collision

When applying differential cryptanalysis to a hash function, a collision for the hash function corresponds to a pair of inputs with output difference zero.

Definition

An i-th-order differential collision for $f\colon S \to T$ is an i-tuple $(a_1, a_2, \ldots, a_i)$ together with a value y such that

$$\Delta_{(a_1,\ldots,a_i)} f(y) = 0\,.$$

Higher-Order Differential Collision

When applying differential cryptanalysis to a hash function, a collision for the hash function corresponds to a pair of inputs with output difference zero.

Definition

An i-th-order differential collision for $f\colon S \rightarrow T$ is an i-tuple $(a_1, a_2, \ldots, a_i)$ together with a value y such that

$$\Delta_{(a_1,\ldots,a_i)} f(y) = 0\,.$$

Note that the common definition of a collision for hash functions corresponds to a higher-order differential collision of order $i = 1$.

Complexity

- What is the *query complexity* of a differential collision of order i?
- From the definition before we see that we can freely choose $i+1$ of the input parameters which then fix the remaining ones

$\Rightarrow$ Query complexity: $\approx 2^{n/(i+1)}$

Complexity

- What is the *query complexity* of a differential collision of order i?
- From the definition before we see that we can freely choose $i+1$ of the input parameters which then fix the remaining ones

$\Rightarrow$ Query complexity: $\approx 2^{n/(i+1)}$

Note that the complexity might be much higher in practice than this bound for the query complexity.

Higher-Order Differential Collision for Block Cipher based Compression Functions

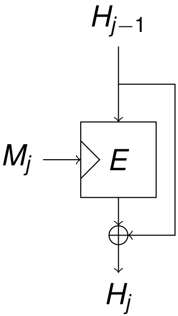
Observation

For any block-cipher-based compression function with which can be written in the form

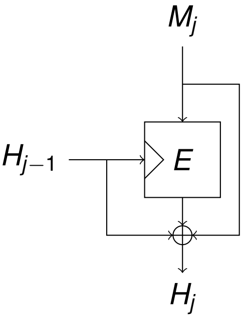
$$f(y) = E(y) + L(y),$$

where L is a linear function with respect to $+$, an i-th-order differential collision for the block cipher transfers to an i-th-order collision for the compression function for $i \geq 2$.

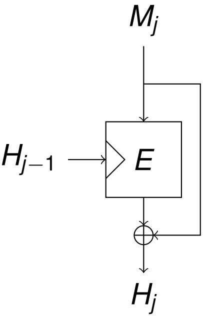
Compression Function Constructions



Davies-Meyer



Miyaguchi-Preneel



Matyas-Meyer-Oseas

Second-order Differential Collision

- Second-order differential collision:

$$f(y) - f(y + a_2) + f(y + a_1 + a_2) - f(y + a_1) = 0$$

Second-order Differential Collision

- Second-order differential collision:

$$f(y) - f(y + a_2) + f(y + a_1 + a_2) - f(y + a_1) = 0$$

- Query complexity: $2^{n/3}$

Second-order Differential Collision

- Second-order differential collision:

$$f(y) - f(y + a_2) + f(y + a_1 + a_2) - f(y + a_1) = 0$$

- Query complexity: $2^{n/3}$
- We are not aware of any algorithm faster than $2^{n/2}$

Basic Attack Strategy

- Split underlying block cipher E into two subparts, $E = E_1 \circ E_0$.
- Assume we are given two differentials for the two subparts:

$$E_0^{-1}(y + \beta) - E_0^{-1}(y) = \alpha \tag{1}$$

and

$$E_1(y + \gamma) - E_1(y) = \delta \tag{2}$$

where the differential in E_0^{-1} holds with probability p_0 and in E_1 holds with probability p_1.

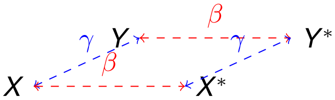
Basic Attack Strategy

- Choose a random value for X.

X

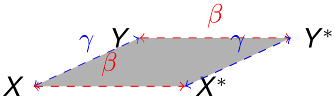
Basic Attack Strategy

- Choose a random value for X.
- Compute $X^* = X + \beta$,
  $Y = X + \gamma$, and $Y^* = X^* + \gamma$.



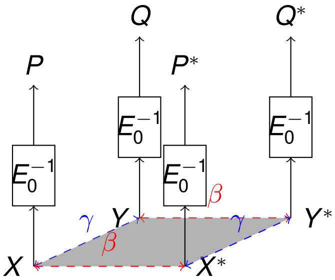
Basic Attack Strategy

- Choose a random value for X.
- Compute $X^* = X + \beta$,
  $Y = X + \gamma$, and $Y^* = X^* + \gamma$.



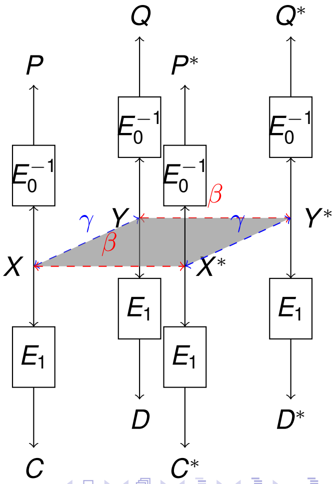
Basic Attack Strategy

- Choose a random value for X.
- Compute $X^* = X + \beta$,
  $Y = X + \gamma$, and $Y^* = X^* + \gamma$.
- Compute backward to obtain
  P, P^*, Q, Q^*.



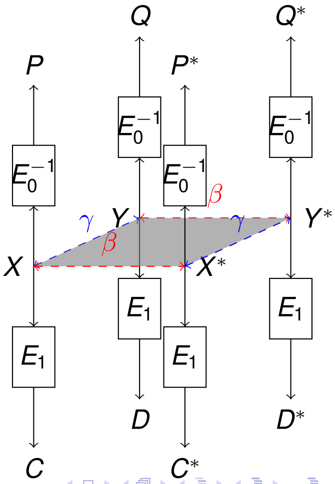
Basic Attack Strategy

- Choose a random value for X.
- Compute $X^* = X + \beta$, $Y = X + \gamma$, and $Y^* = X^* + \gamma$.
- Compute backward to obtain P, P^*, Q, Q^*.
- Compute forward to obtain C, C^*, D, D^*.



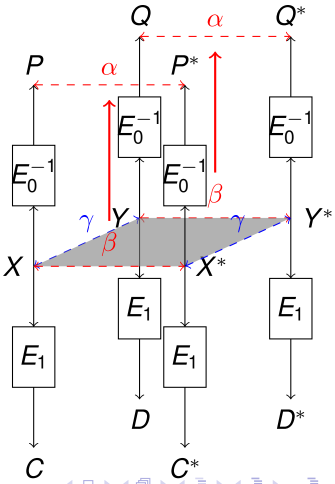
Basic Attack Strategy

- Choose a random value for X.
- Compute $X^* = X + \beta$, $Y = X + \gamma$, and $Y^* = X^* + \gamma$.
- Compute backward to obtain P, P^*, Q, Q^*.
- Compute forward to obtain C, C^*, D, D^*.
- Check if $P^* - P = Q^* - Q$ and $D - C = D^* - C^*$ is fulfilled.



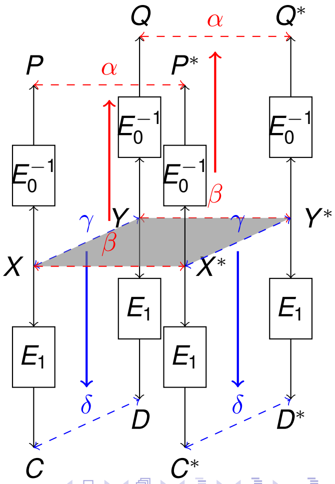
Basic Attack Strategy

- Choose a random value for X.
- Compute $X^* = X + \beta$, $Y = X + \gamma$, and $Y^* = X^* + \gamma$.
- Compute backward to obtain P, P^*, Q, Q^*.
- Compute forward to obtain C, C^*, D, D^*.
- Check if $P^* - P = Q^* - Q$ and $D - C = D^* - C^*$ is fulfilled.



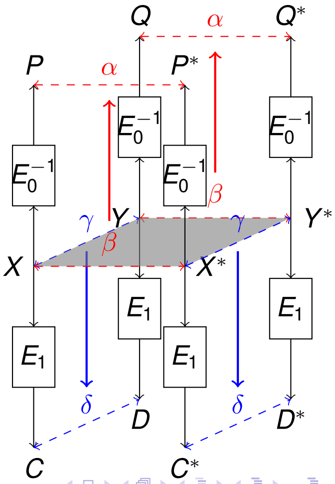
Basic Attack Strategy

- Choose a random value for X.
- Compute $X^* = X + \beta$, $Y = X + \gamma$, and $Y^* = X^* + \gamma$.
- Compute backward to obtain P, P^*, Q, Q^*.
- Compute forward to obtain C, C^*, D, D^*.
- Check if $P^* - P = Q^* - Q$ and $D - C = D^* - C^*$ is fulfilled.



Basic Attack Strategy

- Choose a random value for X.
- Compute $X^* = X + \beta$, $Y = X + \gamma$, and $Y^* = X^* + \gamma$.
- Compute backward to obtain P, P^*, Q, Q^*.
- Compute forward to obtain C, C^*, D, D^*.
- Check if $P^* - P = Q^* - Q$ and $D - C = D^* - C^*$ is fulfilled.
- Attack succeeds with probability $p_0^2 \cdot p_1^2$.



Related Work

Block Cipher Cryptanalysis

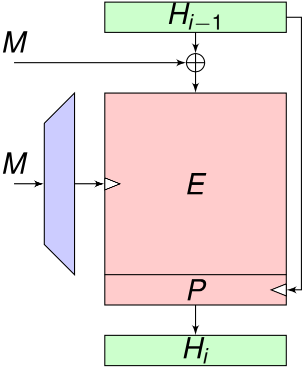
- It stands between the *boomerang attack* and the *inside-out* attack both introduced by Wagner [Wag99]

Hash Functions Cryptanalysis

- A previous application of the boomerang attack to hash functions is due to Joux and Peyrin [JP07]
- The attack bears resemblance with the *rebound attack* introduced by Mendel et al. [MRST09]
- A framework similar to this was independently proposed by Biryukov et al. [BNR11]

Outline

Application to SIMD-512 Permutation



Application to SIMD-512 Permutation

- Second-order differential collision with complexity $\approx 2^{226.52}$

Application to SIMD-512 Permutation

- Second-order differential collision with complexity $\approx 2^{226.52}$
- Finding the differential characteristics for backward and forward direction is the most difficult part of the attack

Application to SIMD-512 Permutation

- Second-order differential collision with complexity $\approx 2^{226.52}$
- Finding the differential characteristics for backward and forward direction is the most difficult part of the attack
- We have two requirements for the differential characteristics:
  - independent
  - high probability

Finding Differential Characteristics

- Linearize the hash function

Finding Differential Characteristics

- Linearize the hash function
  - Modular additions $\rightarrow$ XOR operation
  - Boolean functions f_{IF}, f_{MAJ} $\rightarrow$ 0-function

Finding Differential Characteristics

- Linearize the hash function
  - Modular additions → XOR operation
  - Boolean functions f_{IF}, f_{MAJ} → 0-function
- Use a probabilistic algorithm from coding theory

Finding Differential Characteristics

- Linearize the hash function
  - Modular additions $\rightarrow$ XOR operation
  - Boolean functions f_{IF}, f_{MAJ} $\rightarrow$ 0-function
- Use a probabilistic algorithm from coding theory
- Results
  - Backward: steps 1-18 (probability $2^{-72.04}$)
  - Forward: steps 19-32 (probability $2^{-51.4}$)

Complexity of the Attack

Probability of the Characteristics

- Backward: $2^{-72.04}$
- Forward: $2^{-51.4}$

Complexity of the Attack

Probability of the Characteristics

- Backward: $2^{-72.04}$
- Forward: $2^{-51.4}$

$\Rightarrow$ complexity for the attack is $2^{2\cdot(72.04+51.4)} \approx 2^{247}$

Complexity of the Attack

Probability of the Characteristics

- Backward: $2^{-72.04}$
- Forward: $2^{-51.4}$

$\Rightarrow$ complexity for the attack is $2^{2 \cdot (72.04+51.4)} \approx 2^{247}$

- Ignoring conditions at the end [WYY05]

Complexity of the Attack

Probability of the Characteristics

- Backward: $2^{-72.04}$
- Forward: $2^{-51.4}$

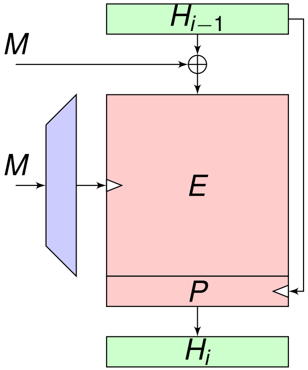
$\Rightarrow$ complexity for the attack is $2^{2\cdot(72.04+51.4)} \approx 2^{247}$

- Ignoring conditions at the end [WYY05]

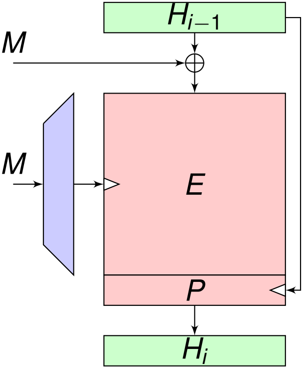
$\Rightarrow$ improved complexity is $2^{226.52}$

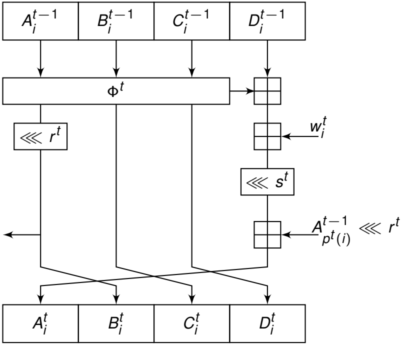
Outline

Extending the Attack to the Compression Function

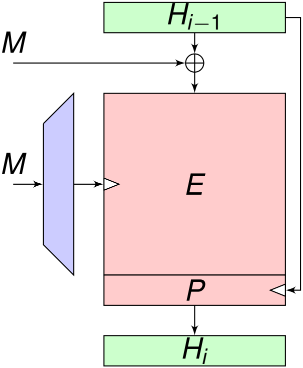


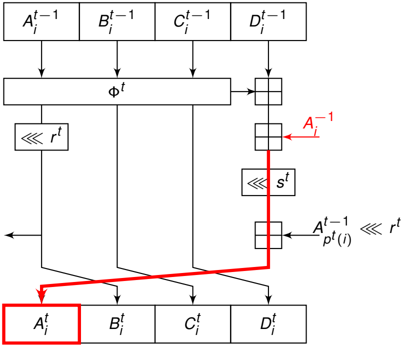
Extending the Attack to the Compression Function



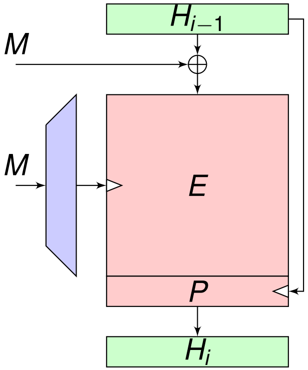


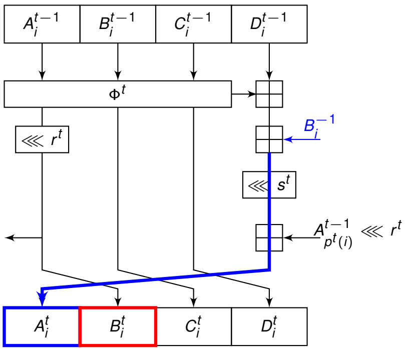
Extending the Attack to the Compression Function



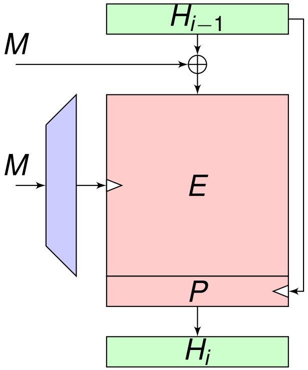


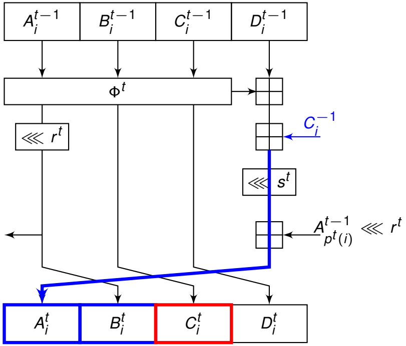
Extending the Attack to the Compression Function



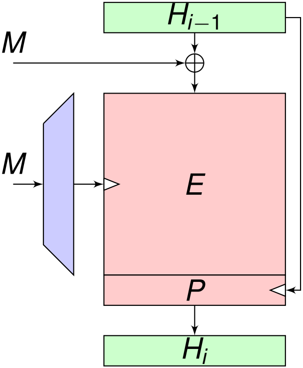


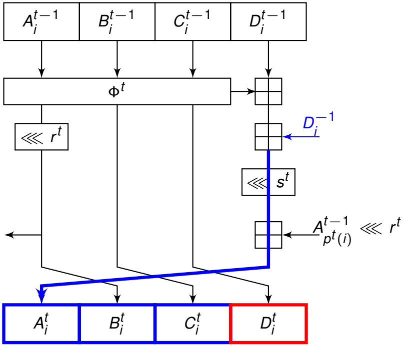
Extending the Attack to the Compression Function



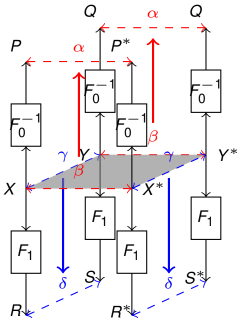


Extending the Attack to the Compression Function

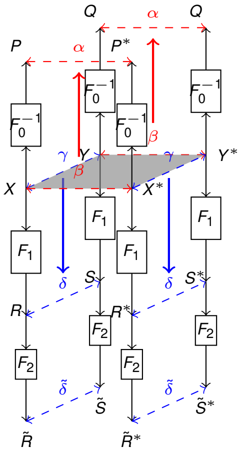




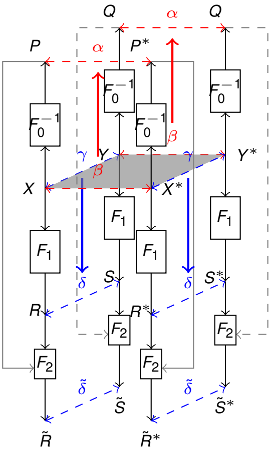
Extended Attack Strategy



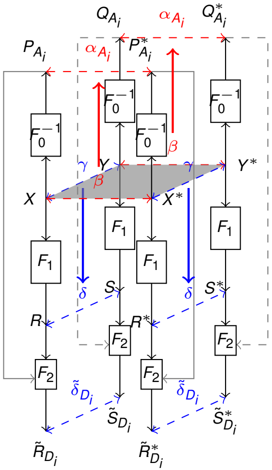
Extended Attack Strategy



Extended Attack Strategy



Extended Attack Strategy



Complexity of the Attack

- Using the same differential characteristic (fix β, γ)
- Backward: only difference in ΔA_6^{-1}
- Forward: only difference in ΔA_3^{31} and ΔB_0^{31}

Complexity of the Attack

- Using the same differential characteristic (fix β, γ)
- Backward: only difference in ΔA_6^{-1}
- Forward: only difference in ΔA_3^{31} and ΔB_0^{31}
  - Input to IF function
  - Used to compute ΔA_6^{32}

Complexity of the Attack

- Using the same differential characteristic (fix β, γ)
- Backward: only difference in ΔA_6^{-1}
- Forward: only difference in ΔA_3^{31} and ΔB_0^{31}
  - Input to IF function
  - Used to compute ΔA_6^{32}
  - Added costs: 2^3
  - Ignore costs: last three steps in both directions

Complexity of the Attack

- Using the same differential characteristic (fix β, γ)
- Backward: only difference in ΔA_6^{-1}
- Forward: only difference in ΔA_3^{31} and ΔB_0^{31}
  - Input to IF function
  - Used to compute ΔA_6^{32}
  - Added costs: 2^3
  - Ignore costs: last three steps in both directions
- Final complexity: $\approx 2^{200.6}$
- Generic complexity: 2^{256}

Outline

Conclusions

- Application of the boomerang attack on SIMD-512
- Using techniques from coding theory to search for two differential characteristics
- Construct a second-order differential collision and define a distinguishing property
- Distinguisher for the full permutation of SIMD-512
- Extend the attack to the full compression function of SIMD-512
  - Best distinguishing attack for SIMD-512 ($2^{200.6}$ vs. 2^{398})

Thank you for your Attention!

Questions?

References I

- Christina Boura and Anne Canteaut.
  Zero-Sum Distinguishers for Iterated Permutations and Application to Keccak- and Hamsi-256.
  In Alex Biryukov, Guang Gong, and Douglas R. Stinson, editors, *Selected Areas in Cryptography*, volume 6544 of *LNCS*, pages 1–17. Springer, 2010.
- Alex Biryukov, Mario Lamberger, Florian Mendel, and Ivica Nikolic.
  Second-Order Differential Collisions for Reduced SHA-256.
  In *ASIACRYPT*, 2011.
  To appear.
- Alex Biryukov, Ivica Nikolic, and Arnab Roy.
  Boomerang Attacks on BLAKE-32.
  In Antoine Joux, editor, *FSE*, volume 6733 of *LNCS*, pages 218–237. Springer, 2011.
- Przemyslaw Sokolowski Ivica Nikolić, Josef Pieprzyk and Ron Steinfeld.
  Rotational Cryptanalysis of (Modified) Versions of BMW and SIMD.
  Available online, 2010.

References II

Antoine Joux and Thomas Peyrin.
Hash Functions and the (Amplified) Boomerang Attack.
In Alfred Menezes, editor, *CRYPTO*, volume 4622 of *LNCS*, pages 244–263. Springer, 2007.

Lars R. Knudsen.
Truncated and Higher Order Differentials.
In Bart Preneel, editor, *FSE*, volume 1008 of *LNCS*, pages 196–211. Springer, 1994.

Xuejia Lai.
Higher Order Derivatives and Differential Cryptanalysis.
In Richard E. Blahut, Daniel J. Costello Jr., Ueli Maurer, and Thomas Mittelholzer, editors, *Communications and Cryptography: Two Sides of One Tapestry*, pages 227–233. Kluwer Academic Publishers, 1994.

References III

- Gaëtan Leurent, Charles Bouillaguet, and Pierre-Alain Fouque.
  SIMD Is a Message Digest.
  Submission to NIST (Round 1), December 2008.
  Available online: http://csrc.nist.gov/groups/ST/hash/sha-3/Round1/submissions_rnd1.html.
- Florian Mendel and Tomislav Nad.
  A Distinguisher for the Compression Function of SIMD-512.
  In Bimal K. Roy and Nicolas Sendrier, editors, *INDOCRYPT*, volume 5922 of *LNCS*, pages 219–232. Springer, 2009.
- Florian Mendel, Christian Rechberger, Martin Schläffer, and Søren S. Thomsen.
  The Rebound Attack: Cryptanalysis of Reduced Whirlpool and Grøstl.
  In Orr Dunkelman, editor, *FSE*, volume 5665 of *LNCS*, pages 260–276. Springer, 2009.
- National Institute of Standards and Technology.
  Cryptographic Hash Algorithm Competition, November 2007.
  Available online: http://csrc.nist.gov/groups/ST/hash/sha-3/index.html.

References IV

David Wagner.
The Boomerang Attack.
In Lars R. Knudsen, editor, *FSE*, volume 1636 of *LNCS*, pages 156–170. Springer, 1999.

Dai Watanabe, Yasuo Hatano, Tsuyoshi Yamada, and Toshinobu Kaneko.
Higher Order Differential Attack on Step-Reduced Variants of *Luffa* v1.
In Seokhie Hong and Tetsu Iwata, editors, *FSE*, volume 6147 of *LNCS*, pages 270–285. Springer, 2010.

Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu.
Finding Collisions in the Full SHA-1.
In Victor Shoup, editor, *CRYPTO*, volume 3621 of *LNCS*, pages 17–36. Springer, 2005.

Hongbo Yu and Xiaoyun Wang.
Cryptanalysis of the Compression Function of SIMD.
In Udaya Parampalli and Philip Hawkes, editors, *ACISP*, volume 6812 of *LNCS*, pages 157–171. Springer, 2011.