Cryptanalysis of WIDEA

Gaëtan Leurent

UCL Crypto Group

FSE 2013

Wide block ciphers

- Most block ciphers have a blocksize of 128 bits
  - 64 bits for lightweight
- Sometimes a larger blocksize is useful
  - More than 2^{64} data with a single key
  - Large key, very high security
  - Hash function design

Wide block ciphers

- Rijndael: 192/256
- Threefish: 256/512/1024
- WIDEA: 256/512

WIDEA

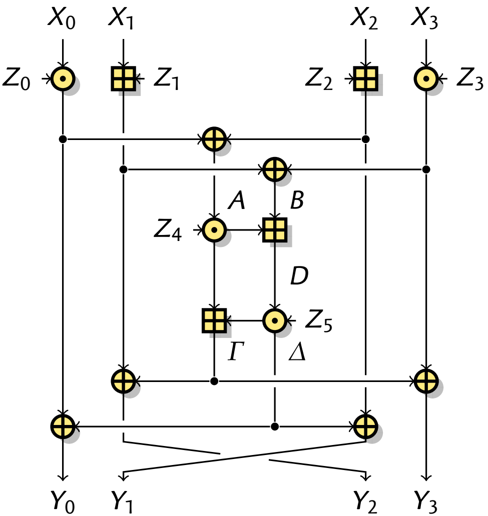
- Wide block cipher based on IDEA
- Designed by Junod and Macchetti [FSE '09]
- Motivation: build a hash function

- Expected to inherit the security of IDEA
  - Full diffusion after one round
  - Mix incompatible operations: ⊞, ⊕, ⊙, ⊗
  - Same number of rounds: 8.5

Previous results

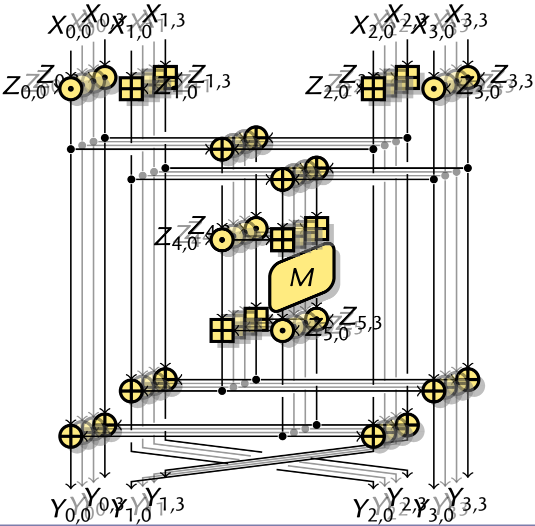
- Weak keys [Nakahara, CANS '12], [Mendel & *al.*, CT-RSA '13]
- Free-start collision (practical) [Mendel & *al.*, CT-RSA '13]

IDEA



- Lai & Massey 1991
- 16-bit words
- 64-bit block, 128-bit key
- 8.5 rounds
- Based on incompatible operations:
  - $\boxplus$: modular addition
  - $\oplus$: bitwise xor
  - $\odot$: mult. mod $2^{16}+1$
- Unbroken after 20^{+} years
  - Weak-keys problems

WIDEA



- Junod & Macchetti 2009
- WIDEA-w: w parallel IDEA
- MDS matrix for diffusion across the slices
  - WIDEA-4: 256-bit block, 512-bit key
  - WIDEA-8: 512-bit block, 1024-bit key
- Efficient SIMD implem.
  - w 16-bit words

WIDEA

- Wide block cipher based on IDEA
- Designed by Junod and Macchetti [FSE '09]
- Motivation: build a hash function

- Expected to inherit the security of IDEA
  - Full diffusion after one round
  - Mix incompatible operations: $\boxplus$, $\oplus$, $\odot$, $\otimes$
  - Same number of rounds: 8.5

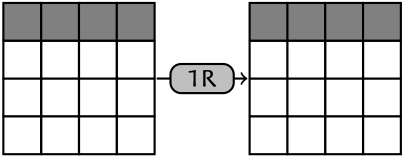
Previous results

- Weak keys [Nakahara, CANS '12], [Mendel *& al.*, CT-RSA '13]
- Free-start collision (practical) [Mendel *& al.*, CT-RSA '13]

Outline

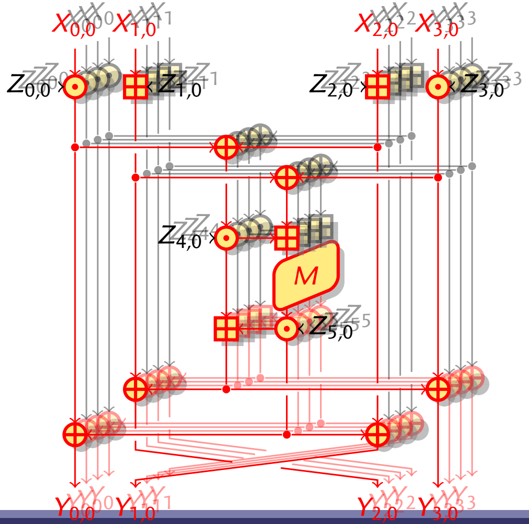
Main idea

- Consider differential attack.
- Can we keep a single slice active?



- Inside the MAD box:

Truncated differential trail



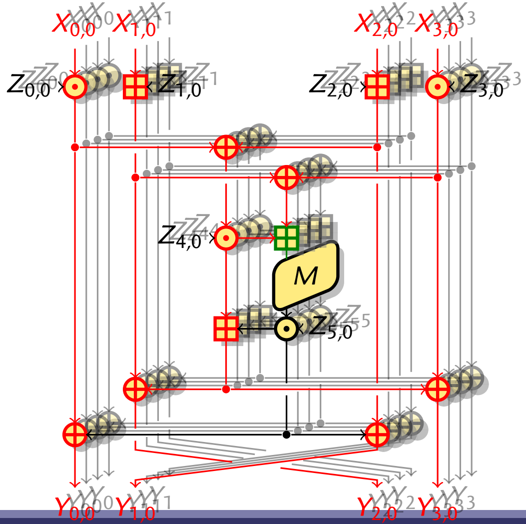
- One input slice active

$$X_{i,0} \neq X'_{i,0}$$
$$X_{i,j} = X_{i,j}$$

- Zero difference at the input of the MDS with probability 2^{-16}
- No effect on other slices

$$Y_{i,0} \neq Y'_{i,0}$$
$$Y_{i,j} = Y_{i,j}$$

Truncated differential trail



- One input slice active

$$X_{i,0} \neq X'_{i,0}$$
$$X_{i,j} = X_{i,j}$$

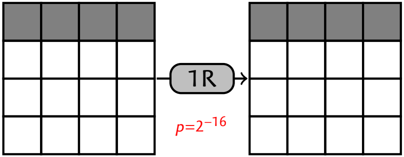
- Zero difference at the input of the MDS with probability 2^{-16}

- No effect on other slices

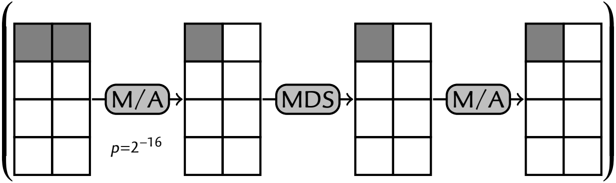
$$Y_{i,0} \neq Y'_{i,0}$$
$$Y_{i,j} = Y_{i,j}$$

Main idea

- Consider differential attack.
- Can we keep a single slice active?

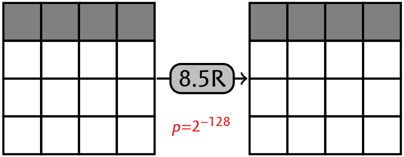


- Inside the MAD box:

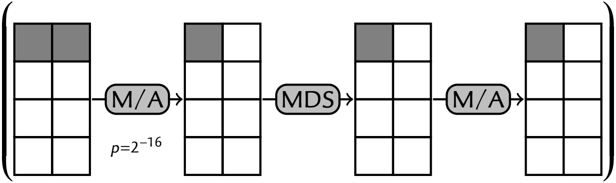


Main idea

- Consider differential attack.
- Can we keep a single slice active?

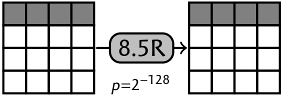


- Inside the MAD box:

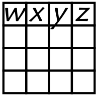


Finding good pairs

- Truncated trail for full 8.5 rounds:



- Use a structure of 2^{64} plaintexts
  - 2^{64} values for one slice
  - Fixed value for the other slices



- 2^{127} candidate pairs with one active slice $\big((w,x,y,z),(w',x',y',z')\big)$
  - One good pair with two structures
  - Look for collisions in inactive slices
- Distinguisher with complexity 2^{65} (succes rate 63%)
  - Strong filtering: no wrong pairs, can break more than 8 rounds

Outline

Using right pairs: first round

Extract key information form right pairs:

- Denote the MDS input as D
- A right pair gives $D = D'$

$$D = \Big(\big((X_0 \odot Z_0) \oplus (X_2 \boxplus Z_2)\big) \odot Z_4\Big) \boxplus \big((X_1 \boxplus Z_1) \oplus (X_3 \odot Z_3)\big)$$

$$D' = \Big(\big((X'_0 \odot Z_0) \oplus (X'_2 \boxplus Z_2)\big) \odot Z_4\Big) \boxplus \big((X'_1 \boxplus Z_1) \oplus (X'_3 \odot Z_3)\big)$$

- Filtering Z_0, Z_1, Z_2, Z_3, Z_4
- 5 pairs should be enough
- Experimental results: need 8 pair
- One bit cannot be recovered (linear): MSB of Z_1

Filtering

Filtering: $D = D'$

$$\Big(\big((X_0 \odot Z_0) \oplus (X_2 \boxplus Z_2)\big) \odot Z_4\Big) \boxplus \big((X_1 \boxplus Z_1) \oplus (X_3 \odot Z_3)\big)$$
$$= \Big(\big((X'_0 \odot Z_0) \oplus (X'_2 \boxplus Z_2)\big) \odot Z_4\Big) \boxplus \big((X'_1 \boxplus Z_1) \oplus (X'_3 \odot Z_3)\big)$$

Meet-in-the-middle:

- Compute $F(X, X', Z_0, Z_2, Z_4)$ for all Z_0, Z_2, Z_4
- Compute $G(X, X', Z_1, Z_3)$ for all Z_1, Z_3
- Find matches

- Complexity: 2^{48}

Filtering

Filtering: $D = D'$

$$\Big(\big((X_0 \odot Z_0) \oplus (X_2 \boxplus Z_2)\big) \odot Z_4\Big) \boxminus \Big(\big((X'_0 \odot Z_0) \oplus (X'_2 \boxplus Z_2)\big) \odot Z_4\Big)$$
$$= \big((X'_1 \boxplus Z_1) \oplus (X'_3 \odot Z_3)\big) \boxminus \big((X_1 \boxplus Z_1) \oplus (X_3 \odot Z_3)\big)$$

Meet-in-the-middle:

- Compute $F(X, X', Z_0, Z_2, Z_4)$ for all Z_0, Z_2, Z_4
- Compute $G(X, X', Z_1, Z_3)$ for all Z_1, Z_3
- Find matches

- Complexity: 2^{48}

Filtering

Filtering: $D = D'$

$$F(X, X', Z_0, Z_2, Z_4) = G(X, X', Z_1, Z_3)$$

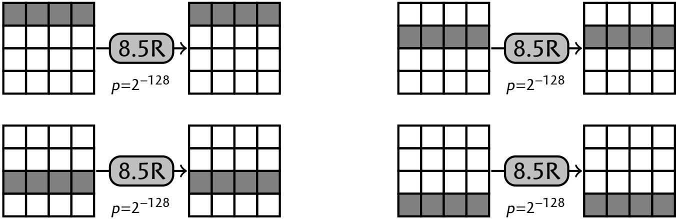
Meet-in-the-middle:

- Compute $F(X, X', Z_0, Z_2, Z_4)$ for all Z_0, Z_2, Z_4
- Compute $G(X, X', Z_1, Z_3)$ for all Z_1, Z_3
- Find matches

- Complexity: 2^{48}

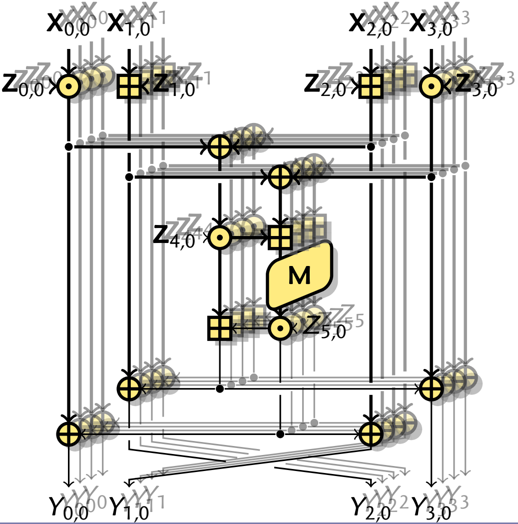
Recovering the full first round key

- Use a trail for each slice:



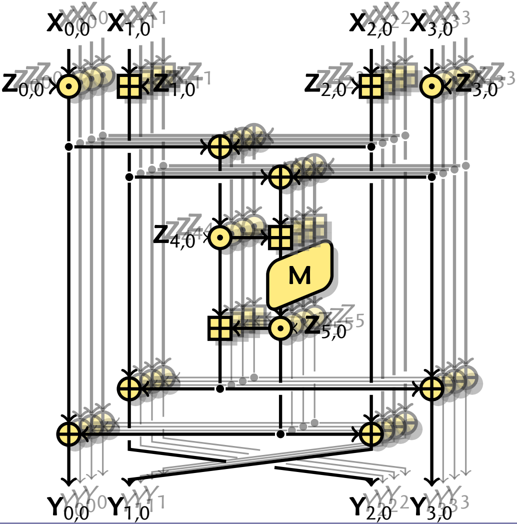
- Attack each slice independantly.
- Recover $Z_{0,i}, Z_{1,i}, Z_{2,i}, Z_{3,i}, Z_{4,i}$.
  - Complexity: $w \cdot 2^{48}$

Second round



- Guess w missing key bits (MSB of Z_1)
- MDS input known (all slices)
  - Compute output
- Guess Z_5 in one slice
  - Compute input of 2nd round
  - Recover 2nd round key: $Z_6, Z_7, Z_8, Z_9, Z_{10}$
- Complexity: $w \cdot 2^{64+w}$

Second round



- Guess w missing key bits (MSB of Z_1)
- MDS input known (all slices)
  - Compute output
- Guess Z_5 in one slice
  - Compute input of 2nd round
  - Recover 2nd round key: $Z_6, Z_7, Z_8, Z_9, Z_{10}$
- Complexity: $w \cdot 2^{64+w}$

Full key recovery

First step: recover $K_{0\dots4}$

for $0 \le i < w$ **do**
- $T \leftarrow \varnothing$
- **for all** k_1, k_3 **do**
  - $G \leftarrow ||_{j=0}^{k} G_i(X^{(i,j)}, X'^{(i,j)}, k_1, k_3)$
  - $T\{G\} \leftarrow (k_1, k_3)$
- **for all** k_0, k_2, k_4 **do**
  - $F \leftarrow ||_{j=0}^{k} F_i(X^{(i,j)}, X'^{(i,j)}, k_0, k_2, k_4)$
  - **if** $F \in T$ **then**
    - $k_1, k_3 \leftarrow T\{F\}$
    - $K_{0\dots4,i} \leftarrow k_0, k_1, k_2, k_3, k_4$

Full key recovery

Second step: recover $K_{5...10}$

for all $K_{1,i}[15]$ **do**
  for $0 \leq i < w$ **do**
    for all k_5 **do**
      $K_{5,i} \leftarrow k_5$
      for all i, k **do**
        $Y^{i,k} \leftarrow \textsc{Round}(X^{(i,k)}, K),\ Y'^{i,k} \leftarrow \textsc{Round}(X'^{(i,k)}, K)$
      $T \leftarrow \varnothing$
      for all k_1, k_3 **do**
        $G \leftarrow \|_{j=0}^{k} G_i(Y^{(i,j)}, Y'^{(i,j)}, k_1, k_3)$
        $T\{G\} \leftarrow (k_1, k_3)$
      for all k_0, k_2, k_4 **do**
        $F \leftarrow \|_{j=0}^{k} F_i(Y^{(i,j)}, Y'^{(i,j)}, k_0, k_2, k_4)$
        if $F \in T$ **then**
          $k_1, k_3 \leftarrow T\{F\}$
          $K_{6...10,i} \leftarrow k_0, k_1, k_2, k_3, k_4$
          goto next i

Complexity analysis

- Reduce the complexity from $w \cdot 2^{64+w}$ to 2^{68} using a few tricks
- Bottleneck is finding good pairs
  - $8 \cdot w$ pairs needed
  - Data complexity: $w \cdot 2^{68}$

1. Using a hash table:
   - Time $w \cdot 2^{68}$, Mem 2^{64}
2. Store and sort:
   - Time $w \cdot 2^{74}$, Mem 2^{64}
3. Time-memory tradeoff:
   - Time $5w \cdot 2^{68+t/2}$, Mem 2^{64-t}, Adaptive CP

Outline

Introduction

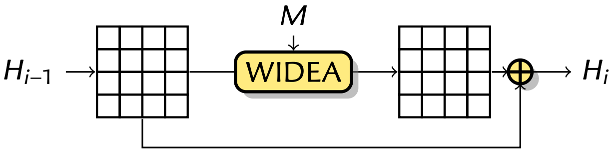
Truncated differential

Key recovery

Hash collisions

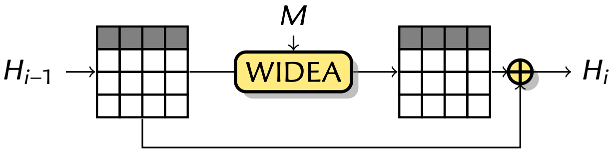
Conclusion

Hash collisions



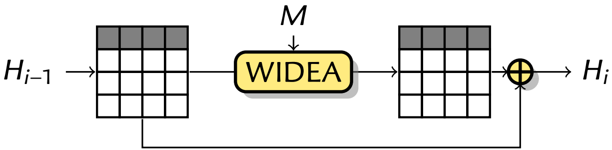
- HIDEA-512 is WIDEA-8 with Davies-Meyer
- Use our truncated differential trail
  1. Find a 448-bit collision H_{i-1}, H'_{i-1}
  2. Hash random message blocks
     - With probability 2^{-128}, the trail is followed
     - With probability 2^{-64}, collision in the feed-forward

Hash collisions



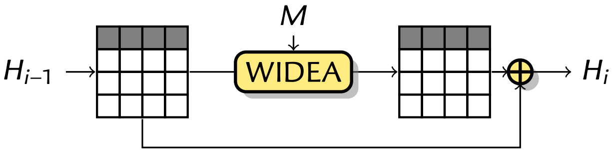
- HIDEA-512 is WIDEA-8 with Davies-Meyer
- Use our truncated differential trail
  1. Find a 448-bit collision H_{i-1}, H'_{i-1}
  2. Hash random message blocks
     - With probability 2^{-128}, the trail is followed
     - With probability 2^{-64}, collision in the feed-forward

Hash collisions



- HIDEA-512 is WIDEA-8 with Davies-Meyer
- Use our truncated differential trail
  1. Find a 448-bit collision H_{i-1}, H'_{i-1}
  2. Hash random message blocks
     - With probability 2^{-128}, the trail is followed
     - With probability 2^{-64}, collision in the feed-forward

Hash collisions



Find P, P' with $T_{448}(H(P)) = T_{448}(H(P'))$ ▷ *Complexity* 2^{224}

repeat

$M \leftarrow Rand()$

until $H(P\|M) = H(P'\|M)$ ▷ *Complexity* 2^{192}

- Full hash function collisions with complexity 2^{224}
  - Very simple attack!
  - Independant of the message expansion.
  - Chosen prefix, meaningful messages, ...

Outline

Summary

Truncated differential trail

- MDS input too small
  - Difference stays in a single IDEA instance with probability 2^{-128}
  - Strong property, can break more than 8 rounds!

1. Key recovery
   - Using structures of 2^{64} plaintext
   - Complexity 2^{70} for WIDEA-4 (256-bit block, 512-bit key)
   - Complexity 2^{71} for WIDEA-8 (512-bit block, 1024-bit key)
2. Hash collisions
   - Complexity 2^{224} for HIDEA-512

Thanks

Questions?

With the support of ERC project CRASH

European Research Council
Established by the European Commission
Supporting top researchers from anywhere in the world