Zero-sum distinguishers

Jean-Philippe Aumasson

For a **permutation family** $\{P_k\}_k$ (e.g., AES-128)

For a **permutation family** $\{P_k\}_k$ (e.g., AES-128)

Standard distinguisher:

- k chosen at random
- attacker makes **black-box** queries to P_k and P_k^{-1}
- attacker returns 0 or 1

For a **permutation family** $\{P_k\}_k$ (e.g., AES-128)

Standard distinguisher:

- k chosen at random
- attacker makes **black-box** queries to P_k and P_k^{-1}
- attacker returns 0 or 1

Known-key distinguisher:

- k chosen at random
- attacker analyzes algorithms of P_k and of P_k^{-1} (**white-box**)
- attacker returns $x_1, \ldots, x_N$ such that

$$\mathcal{R}(x_1, \ldots, x_N, P_k(x_1), \ldots, P_k(x_N)) = 1$$

for some nontrivial relation $\mathcal{R}$

For a **permutation** *P* (e.g., AES-128 without AddRoundKey)

For a **permutation** P (e.g., AES-128 without AddRoundKey)

No-key distinguisher:

- attacker analyzes algorithms of P and of P^{-1} (**white-box**)
- attacker returns $x_1, \ldots, x_N$ such that

$$\mathcal{R}(x_1, \ldots, x_N, P(x_1), \ldots, P(x_N)) = 1$$

for some nontrivial relation $\mathcal{R}$

For a **permutation** P (e.g., AES-128 without AddRoundKey)

No-key distinguisher:

- attacker analyzes algorithms of P and of P^{-1} (**white-box**)
- attacker returns $x_1, \ldots, x_N$ such that

$$\mathcal{R}(x_1, \ldots, x_N, P(x_1), \ldots, P(x_N)) = 1$$

for some nontrivial relation $\mathcal{R}$

Zero-sum distinguisher = no-key distinguisher where

$$\mathcal{R}(x_1, \ldots, x_N, P(x_1), \ldots, P(x_N)) = 1$$

iff

$$\bigoplus_{i=1}^{N} x_i = \bigoplus_{i=1}^{N} P(x_i) = 0$$

How to find zero-sums $(x_1, \ldots, x_N)$?

Case study: the permutation of **Keccak** (SHA-3 candidate)

- 1600-bit state
- 18 nonidentical rounds

How to find zero-sums $(x_1, \ldots, x_N)$?

Case study: the permutation of **Keccak** (SHA-3 candidate)

- 1600-bit state
- 18 nonidentical rounds
- one round has algebraic degree 2 (wrt GF(2))
- one inverse round has algebraic degree 3

How to find zero-sums $(x_1, \dots, x_N)$?

Case study: the permutation of **Keccak** (SHA-3 candidate)

- 1600-bit state
- 18 nonidentical rounds
- one round has algebraic degree 2 (wrt GF(2))
- one inverse round has algebraic degree 3

10 rounds: degree upper bound $2^{10} = 1024$ (suboptimal)
$\Rightarrow$ high-order differential distinguisher in 2^{1024}

How to find zero-sums $(x_1, \ldots, x_N)$?

Case study: the permutation of **Keccak** (SHA-3 candidate)

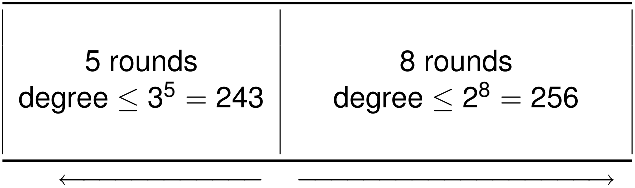
- 1600-bit state
- 18 nonidentical rounds
- one round has algebraic degree 2 (wrt GF(2))
- one inverse round has algebraic degree 3

10 rounds: degree upper bound $2^{10} = 1024$ (suboptimal)
$\Rightarrow$ high-order differential distinguisher in 2^{1024}

13 rounds: degree upper bound $2^{13} \gg 1599$ (optimal)
$\Rightarrow$ ~~high-order differential distinguisher~~

Consider 257 variables in intermediate state after 5 rounds

- preimage = degree-243 mapping
- image = degree-256 mapping
- compute order-257 derivative in both directions



Consider 257 variables in intermediate state after 5 rounds

- preimage = degree-243 mapping
- image = degree-256 mapping
- compute order-257 derivative in both directions

5 rounds degree $\leq 3^5 = 243$	8 rounds degree $\leq 2^8 = 256$

$\longleftarrow$ $\longrightarrow$

Obtain $(x_1, \ldots, x_{2^{257}})$ such that $\bigoplus_{i=1}^{2^{257}} x_i$ is the order-257 derivative of a degre-256 polynomial: must be **zero**

$\Rightarrow$ zero-sum distinguisher on 13 rounds in 2^{257}

Optimizations: exploit structure of the (inverse) permutation

#rounds	complexity
8	2^{17}
10	2^{60}
12	2^{128}
14	2^{256}
16	2^{1024}

(2^{1600} ideally)

(18 rounds in full version)

Security of (reduced) hash function seems unaffected

Application to other SHA-3 candidates:

Q permutation of **Luffa** (256-bit)

- distinguisher on full version (8 rounds) in 2^{81}
- distinguisher on 7 rounds in 2^{27}

P_f permutation of **Hamsi**

- distinguisher on full version (512-bit) in 2^{27}
- distinguisher on full version (1024-bit) in 2^{729}

Does not extend to attacks on hash functions…

Application to (reduced) KATAN and KTANTAN ciphers?