

Differential MITM attacks on SLIM and LBCIoT

and the limits of differential attacks

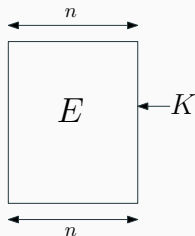
Peter Grochal Martin Stanek
CECC 2025, Budapest, Hungary

Faculty of Mathematics, Physics and Informatics
Comenius University, Bratislava

Cryptanalysis of ciphers

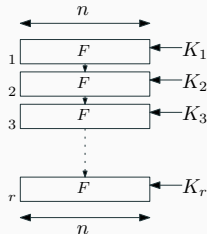
Symmetric block ciphers

- Symmetric n -bit cipher
 - We focus on iterated block ciphers



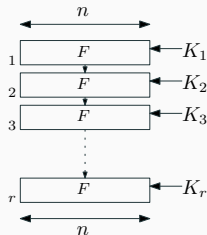
Symmetric block ciphers

- Symmetric n -bit cipher
 - We focus on iterated block ciphers



Lightweight symmetric block ciphers

- Symmetric n -bit cipher
 - We focus on iterated block ciphers

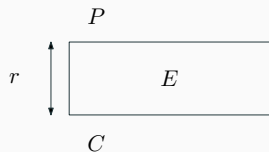


- *Lightweight* cipher
 - Optimize resources as power, memory, hardware complexity...
 - Goal: reach comparable security to classical ciphers.

- Differential MITM cryptanalysis
 - Proposed improvements: *deterministic bits, identical bits*
 - New attacks against SLIM a LBCIoT
 - Best attack to date on LBCIoT
- Problems with differential attacks
 - Low-probability differentials
 - Commonly used assumptions of filter uniformity

Differential MITM cryptanalysis

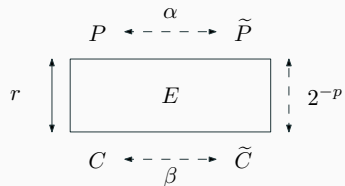
Iterated block cipher



Iterated symmetric block cipher

- with r rounds

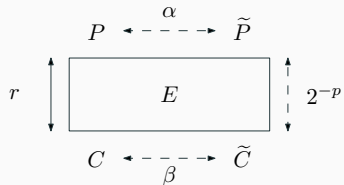
Differential



Differential (α, β) with probability 2^{-p}

$$\alpha = P \oplus \tilde{P} \stackrel{2^{-p}}{\rightsquigarrow} C \oplus \tilde{C} = \beta$$

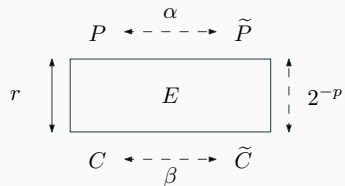
Differential



Differential (α, β) with probability 2^{-p}

$$\alpha = P \oplus \tilde{P} \stackrel{2^{-p}}{\rightsquigarrow} C \oplus \tilde{C} = \beta$$

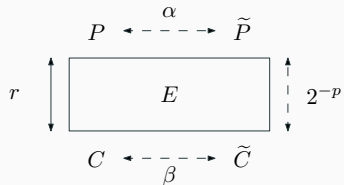
Differential



Differential (α, β) with probability 2^{-p}

$$\alpha = P \oplus \tilde{P} \quad \overset{2^{-p}}{\rightsquigarrow} \quad C \oplus \tilde{C} = \beta$$

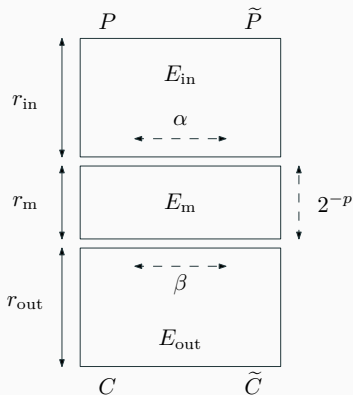
Differential



Differential (α, β) with probability 2^{-p}

$$\alpha = P \oplus \tilde{P} \stackrel{2^{-p}}{\rightsquigarrow} C \oplus \tilde{C} = \beta$$

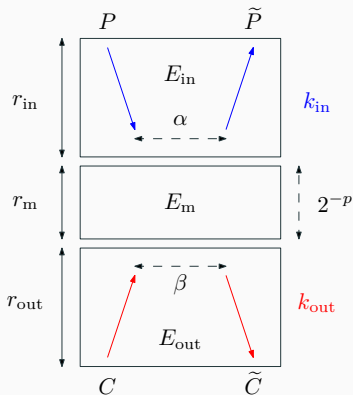
Differential MITM cryptanalysis – $DM(r_{in}, r_m, r_{out})$



Attack CPA

- Cipher is split into: $E_{out} \circ E_m \circ E_{in}$.
- Differential (α, β) over E_m with probability 2^{-p} .
- Returns candidates for k_{in} and k_{out} .
- Boura et al. (CRYPTO 2023)

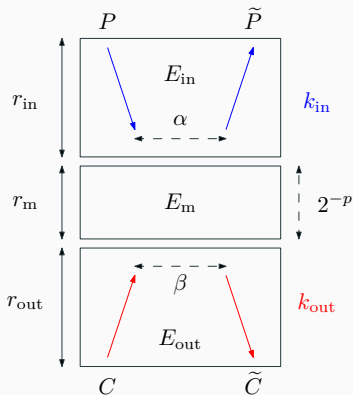
Differential MITM cryptanalysis – $DM(r_{in}, r_m, r_{out})$



Attack CPA

- For each guess i for k_{in} compute \tilde{P}_i
- For each guess j for k_{out} compute \tilde{C}_j
- Collision $E(\tilde{P}_i) == \tilde{C}_j$ leads to a candidate (i, j) .
- Boura et al. (CRYPTO 2023)

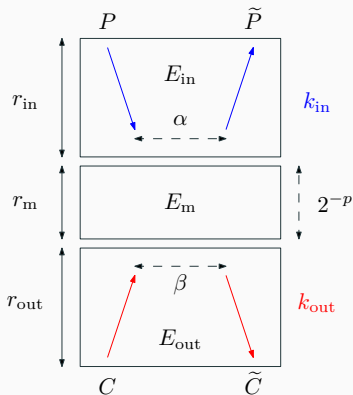
Differential MITM cryptanalysis – $DM(r_{in}, r_m, r_{out})$



Attack CPA

- Cipher is split into: $E_{out} \circ E_m \circ E_{in}$.
- Differential (α, β) over E_m with probability 2^{-p} .
- Returns candidates for k_{in} and k_{out} .
- Boura et al. (CRYPTO 2023)

Differential MITM cryptanalysis – $DM(r_{in}, r_m, r_{out})$



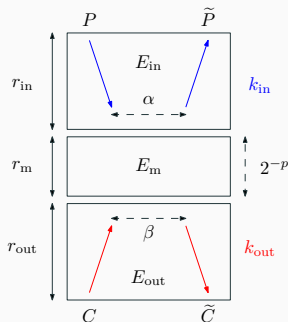
Attack CPA – repeat $\kappa \cdot 2^p$ times

- Cipher is split into: $E_{out} \circ E_m \circ E_{in}$.
- Differential (α, β) over E_m with probability 2^{-p} .
- Returns candidates for k_{in} and k_{out} .
- Boura et al. (CRYPTO 2023)

Proposed improvements

deterministic bits

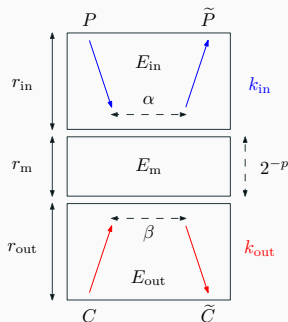
- bits (in \tilde{C}), whose difference (from C) is determinable from β .
- detect incorrect candidates sooner, thus making the attack faster.



Proposed improvements

deterministic bits

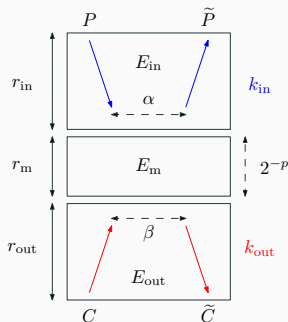
- bits (in \tilde{C}), whose difference (from C) is determinable from β .
- detect incorrect candidates sooner, thus making the attack faster.
- truncated differential with probability 1 over E_{out} .
- set of impossible differentials over E_{out} .



Proposed improvements

deterministic bits

- bits (in \tilde{C}), whose difference (from C) is determinable from β .
- detect incorrect candidates sooner, thus making the attack faster.
- truncated differential with probability 1 over E_{out} .
- set of impossible differentials over E_{out} .



multiplicity of candidates

- count the multiplicity of candidates, output only the most frequent
- significantly lowers the number of candidates

Proposed improvements: Identical bits

identical bit: bit (in k_{in} or k_{out}) whose value is identical across all candidates with the highest multiplicity.

Proposed improvements: Identical bits

identical bit: bit (in k_{in} or k_{out}) whose value is identical across all candidates with the highest multiplicity.

Attack outputs only the indices of identical bits and their values.

```
0000000110011001 }  
0001000110011001 } ~→ 01..100110011001  
0010000110011001 }  
0011000110011001 }
```

Proposed improvements: Identical bits

identical bit: bit (in k_{in} or k_{out}) whose value is identical across all candidates with the highest multiplicity.

Attack outputs only the indices of identical bits and their values.

```
0000000110011001 }  
0001000110011001 }  
0010000110011001 } ~→ 01..10....011001  
0011000110011001 }  
0011001001011001 }
```

Experiments: Identical bits

cipher	r	attack	identical bits	$ k_{in} + k_{out} $
SLIM	8	DM(1, 6, 1)	15.9	19
	8	DM(2, 4, 2)	37.8	42
	10	DM(1, 8, 1)	11.7	17
LBCIoT	12	DM(2, 8, 2)	5.9	6
	14	DM(2, 10, 2)	8.8	9
	18	DM(2, 14, 2)	7.9	10

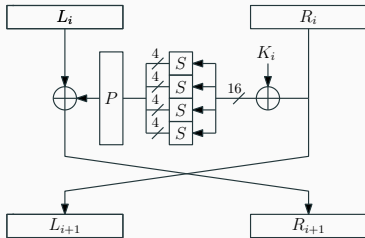
Average number of identical bits

Experiments: Identical bits

cipher	r	attack	identical bits	$ k_{in} + k_{out} $
SLIM	8	DM(1, 6, 1)	15.9	19
	8	DM(2, 4, 2)	37.8	42
	10	DM(1, 8, 1)	11.7	17
LBCIoT	12	DM(2, 8, 2)	5.9	6
	14	DM(2, 10, 2)	8.8	9
	18	DM(2, 14, 2)	7.9	10

Average number of identical bits

Our results: Cryptanalysis of SLIM



- Cryptanalysis of cipher **SLIM**

- Block: 32bit; Key: 80bit

- Rounds: 32

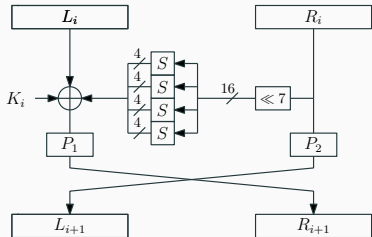
- Best attack: ~~14~~ 19 rounds

- Nobuyuki Sugio

(2024, IET Information Security)

- We propose attack against 18-round SLIM.

Our results: Cryptanalysis of LBCIoT



- Cryptanalysis of cipher **LBCIoT**
 - Block: 32bit; Key: 80bit
 - Rounds: 32
 - Best attack: 19 rounds
- Yen Yee et al. (2023, Heliyon)
- We propose attack against 26-round LBCIoT.

Attacks against SLIM a LBCIoT

cipher	r	attack	$Time$	$Space$	$Data$	$ k_{in} + k_{out} $
SLIM	13*	differential	2^{31}	2^{12}	2^{31}	12
	14*	differential	2^{32}	2^{12}	2^{32}	12
	19	linear	$2^{64.4}$	2^{38}	2^{32}	36
LBCIoT	18	differential†	2^{31}	2^8	2^{31}	16
	19*	differential†	2^{32}	2^3	2^{31}	16

(*) The attack fails for a substantial portion of keys.

(†) The published version of the attack fails. But it can be tweaked to be correct.

Selected attacks on reduced versions of SLIM and LBCIoT

Our results: Attacks against SLIM a LBCIoT

cipher	r	attack	Time	Space	Data	$ k_{in} + k_{out} $
SLIM	13*	differential	2^{31}	2^{12}	2^{31}	12
	14*	differential	2^{32}	2^{12}	2^{32}	12
	16	DM(3, 11, 2)	$\kappa \cdot 2^{71}$	$\kappa \cdot 2^{65}$	2^{32}	71
	18*	DM(3, 13, 2)	$\kappa \cdot 2^{73}$	$\kappa \cdot 2^{67}$	2^{32}	68
	19	linear	$2^{64.4}$	2^{38}	2^{32}	36
LBCIoT	18	differential†	2^{31}	2^8	2^{31}	16
	19*	differential†	2^{32}	2^3	2^{31}	16
	25	DM(4, 17, 4)	$\kappa \cdot 2^{71}$	$\kappa \cdot 2^{71}$	2^{32}	75
	26*	DM(4, 18, 4)	$\kappa \cdot 2^{67}$	$\kappa \cdot 2^{65}$	2^{32}	67

(*) The attack fails for a substantial portion of keys.

(†) The published version of the attack fails. But it can be tweaked to be correct.

Selected attacks on reduced versions of SLIM and LBCIoT

Problems with differential attacks

Low-probability differentials

How does one compute the probability of a differential?

$$\Pr[E_k(x) \oplus E_k(x \oplus \alpha) = \beta ; \text{ for random } x \text{ and random } k].$$

Low-probability differentials

How does one compute the probability of a differential?

$$\Pr[E_k(x) \oplus E_k(x \oplus \alpha) = \beta ; \text{ for random } x \text{ and random } k].$$

Low-probability differentials

How does one compute the probability of a differential?

$\Pr[E_k(x) \oplus E_k(x \oplus \alpha) = \beta ; \text{ for random } x \text{ and random } k]$.

Low-probability differentials

How does one compute the probability of a differential?

$$\Pr[E_k(x) \oplus E_k(x \oplus \alpha) = \beta ; \text{ for random } x \text{ and random } k].$$

We need the differential to occur for the cipher (key) we attack.

Low-probability differentials

How does one compute the probability of a differential?

$$\Pr[E_k(x) \oplus E_k(x \oplus \alpha) = \beta ; \text{ for random } x \text{ and random } k].$$

We need the differential to occur for the cipher (key) we attack.

$$\Pr[E_k(x) \oplus E_k(x \oplus \alpha) = \beta ; \text{ for random } x \text{ and } \boxed{\text{target}} k].$$

Low-probability differentials

How does one compute the probability of a differential?

$\Pr[E_k(x) \oplus E_k(x \oplus \alpha) = \beta ; \text{ for random } x \text{ and } \text{random } k]$.

We need the differential to occur for the cipher (key) we attack.

$\Pr[E_k(x) \oplus E_k(x \oplus \alpha) = \beta ; \text{ for random } x \text{ and } \text{target } k]$.

Low-probability differentials

How does one compute the probability of a differential?

$$\Pr[E_k(x) \oplus E_k(x \oplus \alpha) = \beta ; \text{ for random } x \text{ and } \boxed{\text{random}} k].$$

We need the differential to occur for the cipher (key) we attack.

$$\Pr[E_k(x) \oplus E_k(x \oplus \alpha) = \beta ; \text{ for random } x \text{ and } \boxed{\text{target}} k].$$

Low-probability differential: Differential with probability close to 2^{-n} .

Experiments: Low-probability differentials

r	α	β	2^{-P}	% of keys
11	4827 0080	0020 08b4	2^{-26}	0%
12	0b82 000a	0a00 801b	2^{-28}	30.06%
13	a208 a000	a000 b208	2^{-31}	40.61%

Percentage of tested keys for SLIM for which the differentials **never** occurs

r	α	β	2^{-P}	% of keys
16	0006 0400	0020 1000	2^{-26}	0%
17	0006 0400	0100 2040	2^{-28}	0.01%
18	6000 0040	0000 0800	2^{-30}	5.95%

Percentage of tested keys for LBCIoT for which the differentials **never** occurs

Experiments: Low-probability differentials

r	α	β	2^{-P}	% of keys
11	4827 0080	0020 08b4	2^{-26}	0%
12	0b82 000a	0a00 801b	2^{-28}	30.06%
13	a208 a000	a000 b208	2^{-31}	40.61%

Percentage of tested keys for SLIM for which the differentials **never** occurs

r	α	β	2^{-P}	% of keys
16	0006 0400	0020 1000	2^{-26}	0%
17	0006 0400	0100 2040	2^{-28}	0.01%
18	6000 0040	0000 0800	2^{-30}	5.95%

Percentage of tested keys for LBCIoT for which the differentials **never** occurs

Experiments: Low-probability differentials

r	α	β	2^{-P}	% of keys
11	4827 0080	0020 08b4	2^{-26}	0%
12	0b82 000a	0a00 801b	2^{-28}	30.06%
13	a208 a000	a000 b208	2^{-31}	40.61%

Percentage of tested keys for SLIM for which the differentials **never** occurs

r	α	β	2^{-P}	% of keys
16	0006 0400	0020 1000	2^{-26}	0%
17	0006 0400	0100 2040	2^{-28}	0.01%
18	6000 0040	0000 0800	2^{-30}	5.95%

Percentage of tested keys for LBCIoT for which the differentials **never** occurs

Complexity estimates

$$\mathcal{T} = \overbrace{\kappa \cdot 2^P}^{\text{for each } P} \cdot \left(\underbrace{2^{|k_{\text{in}}|}}_{\text{guess } k_{\text{in}}} + \underbrace{2^{|k_{\text{out}}|}}_{\text{guess } k_{\text{out}}} + \underbrace{2^{|k_{\text{in}}|+|k_{\text{out}}|-n}}_{\text{\#candidates added}} \right)$$

$$\mathcal{S} \leq \underbrace{2^{|k_{\text{in}}|}}_{|H|} + \overbrace{\kappa \cdot 2^P}^{\text{for each } P} \cdot \underbrace{2^{|k_{\text{in}}|+|k_{\text{out}}|-n}}_{\text{\#candidates added}}$$

Complexity estimates

$$\mathcal{T} = \overbrace{\kappa \cdot 2^p}^{\text{for each } P} \cdot \left(\underbrace{2^{|k_{\text{in}}|}}_{\text{guess } k_{\text{in}}} + \underbrace{2^{|k_{\text{out}}|}}_{\text{guess } k_{\text{out}}} + \underbrace{2^{|k_{\text{in}}|+|k_{\text{out}}|-n}}_{\text{\#candidates added}} \right)$$

$$\mathcal{S} \leq \underbrace{2^{|k_{\text{in}}|}}_{|H|} + \overbrace{\kappa \cdot 2^p}^{\text{for each } P} \cdot \underbrace{2^{|k_{\text{in}}|+|k_{\text{out}}|-n}}_{\text{\#candidates added}}$$

How does one estimate the number of candidates added?

$$2^{|k_{\text{in}}|+|k_{\text{out}}|} \cdot 2^{-n}$$

Complexity estimates

$$\mathcal{T} = \overbrace{\kappa \cdot 2^p}^{\text{for each } P} \cdot \left(\underbrace{2^{|k_{\text{in}}|}}_{\text{guess } k_{\text{in}}} + \underbrace{2^{|k_{\text{out}}|}}_{\text{guess } k_{\text{out}}} + \underbrace{2^{|k_{\text{in}}|+|k_{\text{out}}|-n}}_{\text{\#candidates added}} \right)$$

$$\mathcal{S} \leq \underbrace{2^{|k_{\text{in}}|}}_{|H|} + \overbrace{\kappa \cdot 2^p}^{\text{for each } P} \cdot \underbrace{2^{|k_{\text{in}}|+|k_{\text{out}}|-n}}_{\text{\#candidates added}}$$

How does one estimate the number of candidates added?

$$2^{|k_{\text{in}}|+|k_{\text{out}}|} \cdot 2^{-n}$$

Complexity estimates

$$\mathcal{T} = \overbrace{\kappa \cdot 2^p}^{\text{for each } P} \cdot \left(\underbrace{2^{|k_{\text{in}}|}}_{\text{guess } k_{\text{in}}} + \underbrace{2^{|k_{\text{out}}|}}_{\text{guess } k_{\text{out}}} + \underbrace{2^{|k_{\text{in}}|+|k_{\text{out}}|-n}}_{\text{\#candidates added}} \right)$$

$$\mathcal{S} \leq \underbrace{2^{|k_{\text{in}}|}}_{|H|} + \overbrace{\kappa \cdot 2^p}^{\text{for each } P} \cdot \underbrace{2^{|k_{\text{in}}|+|k_{\text{out}}|-n}}_{\text{\#candidates added}}$$

How does one estimate the number of candidates added?

$$2^{|k_{\text{in}}|+|k_{\text{out}}|} \cdot 2^{-n}$$

Complexity estimates

$$\mathcal{T} = \overbrace{\kappa \cdot 2^p}^{\text{for each } P} \cdot \left(\underbrace{2^{|k_{\text{in}}|}}_{\text{guess } k_{\text{in}}} + \underbrace{2^{|k_{\text{out}}|}}_{\text{guess } k_{\text{out}}} + \underbrace{2^{|k_{\text{in}}|+|k_{\text{out}}|-n}}_{\text{\#candidates added}} \right)$$

$$\mathcal{S} \leq \underbrace{2^{|k_{\text{in}}|}}_{|H|} + \overbrace{\kappa \cdot 2^p}^{\text{for each } P} \cdot \underbrace{2^{|k_{\text{in}}|+|k_{\text{out}}|-n}}_{\text{\#candidates added}}$$

How does one estimate the number of candidates added?

$$2^{|k_{\text{in}}|+|k_{\text{out}}|} \cdot 2^{-n}$$

$$E(\tilde{P}_i) = \tilde{C}_j$$

Assumptions of filter uniformity

- *filter* is a boolean function
 - Input: variables related to the current candidate,
 - Output: 0 if the candidate can be discarded; 1 otherwise.
- *l-bit filter* is a filter, which for an uniformly random input returns 1 with probability 2^{-l} .

Results: Assumptions of filter uniformity

- *filter* is a boolean function
 - Input: variables related to the current candidate,
 - Output: 0 if the candidate can be discarded; 1 otherwise.
- *l-bit filter* is a filter, which for an uniformly random input returns 1 with probability 2^{-l} .
- Various analyses often implicitly assume uniformly random input.

Results: Assumptions of filter uniformity

- *filter* is a boolean function
 - Input: variables related to the current candidate,
 - Output: 0 if the candidate can be discarded; 1 otherwise.
- *l-bit filter* is a filter, which for an uniformly random input returns 1 with probability 2^{-l} .
- Various analyses often implicitly assume uniformly random input.
- Results of our experimental verification (on nontrivial no. rounds):
 - The assumption was **never** satisfied
 - Theoretical estimates were off (too optimistic).

Experiments: Assumptions of filter uniformity

	r	attack	actual number	expected number
SLIM	8	DM(1, 6, 1)	119	$\kappa \cdot 2^{-1}$
	8	DM(2, 4, 2)	9 102 387	458 752
	10	DM(1, 8, 1)	166	56
LBCIoT	12	DM(2, 8, 2)	22	$\kappa \cdot 2^{-16}$
	14	DM(2, 10, 2)	40	$\kappa \cdot 2^{-9}$
	18	DM(2, 14, 2)	72	$\kappa \cdot 2^0$

Numbers of candidates in an differential MITM attack ($\kappa = 7$).

Experiments: Assumptions of filter uniformity

	r	attack	actual number	expected number
SLIM	8	DM(1, 6, 1)	119	$\kappa \cdot 2^{-1}$
	8	DM(2, 4, 2)	9 102 387	458 752
	10	DM(1, 8, 1)	166	56
LBCIoT	12	DM(2, 8, 2)	22	$\kappa \cdot 2^{-16}$
	14	DM(2, 10, 2)	40	$\kappa \cdot 2^{-9}$
	18	DM(2, 14, 2)	72	$\kappa \cdot 2^0$

Numbers of candidates in an differential MITM attack ($\kappa = 7$).

Experiments: Assumptions of filter uniformity

	r	attack	actual number	expected number
SLIM	8	DM(1, 6, 1)	119	$\kappa \cdot 2^{-1}$
	8	DM(2, 4, 2)	9 102 387	458 752
	10	DM(1, 8, 1)	166	56
LBCIoT	12	DM(2, 8, 2)	22	$\kappa \cdot 2^{-16}$
	14	DM(2, 10, 2)	40	$\kappa \cdot 2^{-9}$
	18	DM(2, 14, 2)	72	$\kappa \cdot 2^0$

Numbers of candidates in an differential MITM attack ($\kappa = 7$).

- Differential MITM cryptanalysis
 - Proposed improvements: *deterministic bits, identical bits*
 - New attacks against SLIM a LBCIoT
 - Best attack to date on LBCIoT
- Problems with differential attacks
 - Low-probability differentials
 - Commonly used assumptions of filter uniformity

