Improving impossible differential cryptanalysis

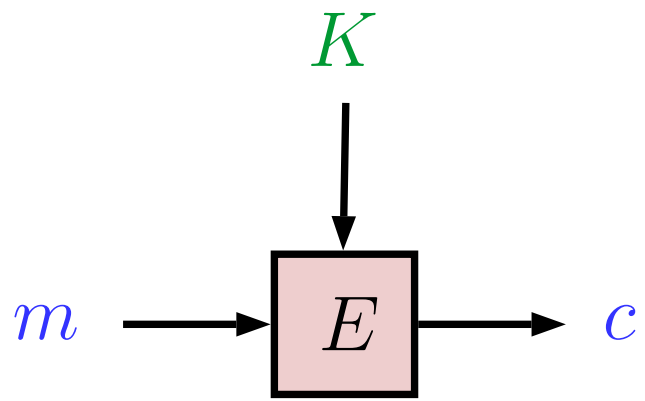
Christina Boura

joint work with Virginie Lallemand, María Naya-Plasencia and Valentin Suder

Séminaire CCA, June 12, 2015

Block ciphers

$$\begin{aligned} E : \{0,1\}^n \times \{0,1\}^k &\rightarrow \{0,1\}^n \\ (m, K) &\mapsto E(m, K) = c \end{aligned}$$



Lightweight block ciphers

- Block ciphers designed for constrained environnements (RFID tags, sensor networks,. . .)
- A high number of proposals in the last few years (PRESENT, CLEFIA, LED, LBlock, Piccolo, TWINE, KLINE, Zorro, PRINTCipher, PRINCE, SIMON, SPECK, . . .)
- New ISO/IEC standards: PRESENT, CLEFIA.

Need to identify the robust algorithms for future use.

Cryptanalysis of block ciphers

In symmetric key cryptography security proofs are partial and insuffient.

The only way to conclude that a block cipher is strong is to evaluate its security against different kind of attacks.

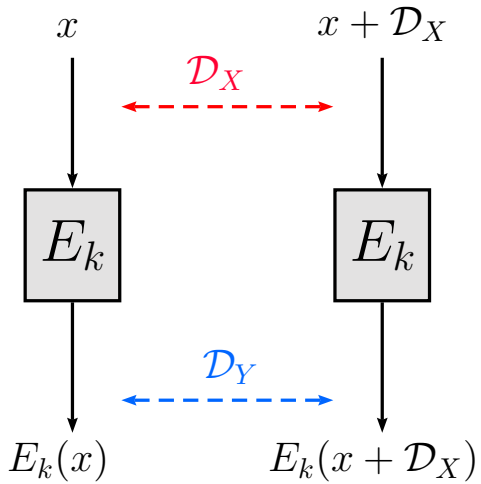
- Generic attack: exhaustive search of the key in $\mathcal{O}(2^k)$.
- A block cipher is considered unbroken as far as no attack faster than exhaustive search exists.

There is a multitude of attacks against block ciphers : differential, linear, algebraic, higher-order differential, integral, impossible differential, ...

Differential cryptanalysis

Introduced by Biham and Shamir in '90.

Given an input difference between two plaintexts, some output differences occur more often than others.



- A differential is a pair $(\mathcal{D}_X, \mathcal{D}_Y)$.

Impossible differential cryptanalysis

Introduced by Knudsen and Biham et al. in '99.

Exploit differentials of probability 0.

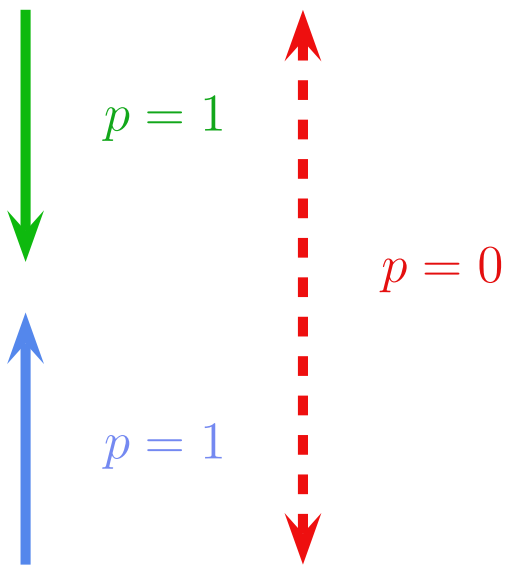
Why care about impossible differential attacks?

- Very powerful attacks (lead to the best cryptanalysis against many ciphers, e.g. some famous Feistel constructions)
- Were for a long time the most successful attacks against the AES.
- Not fully understood and exploited in an non-optimal way due to their high technicality.

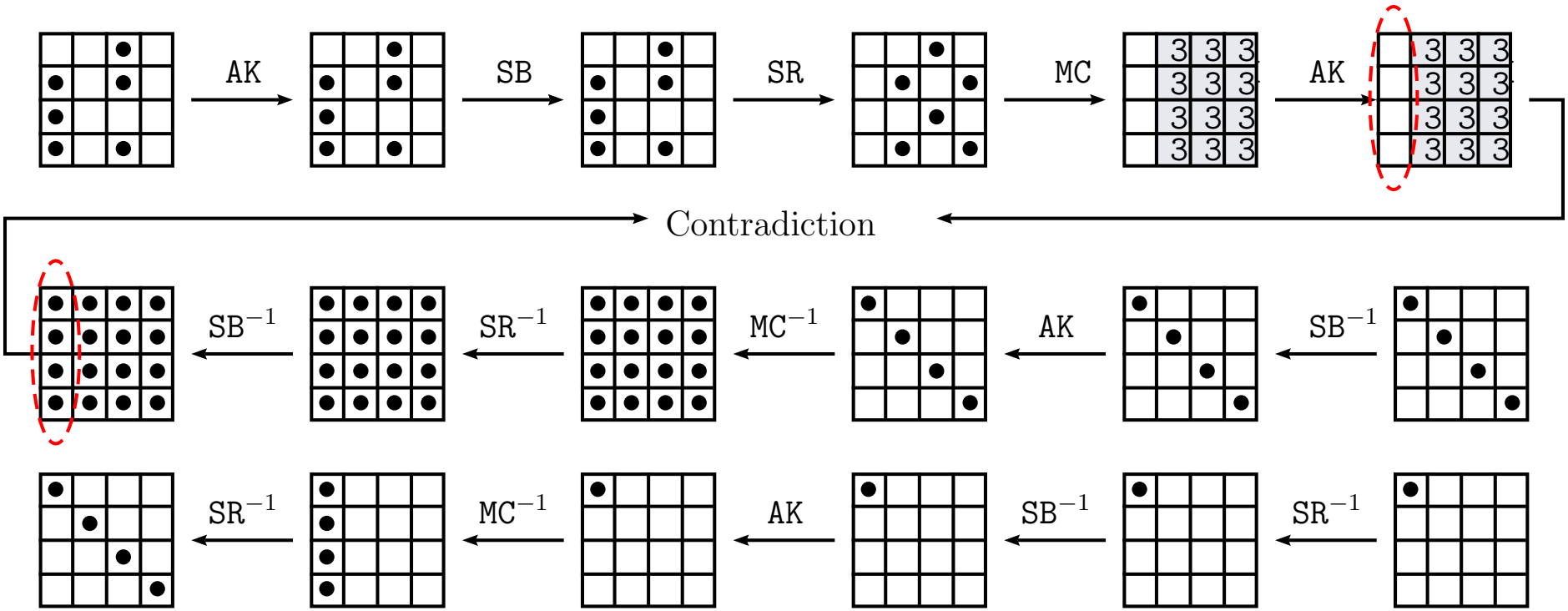
First step: Find an impossible differential

- Well understood and "easy" part of the attack.
- Algorithms for finding impossible differentials on a given cipher exist.

Find the impossible differential using the Miss-in-the-middle technique.



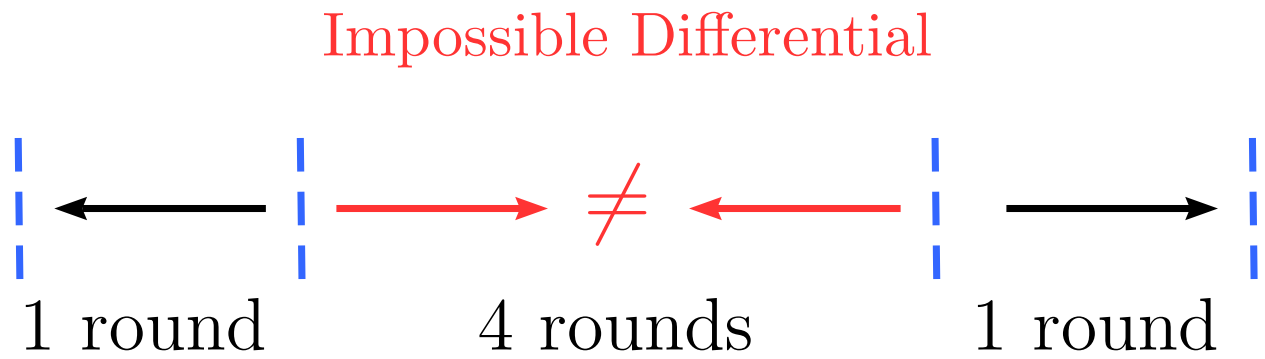
Miss in the Middle for AES



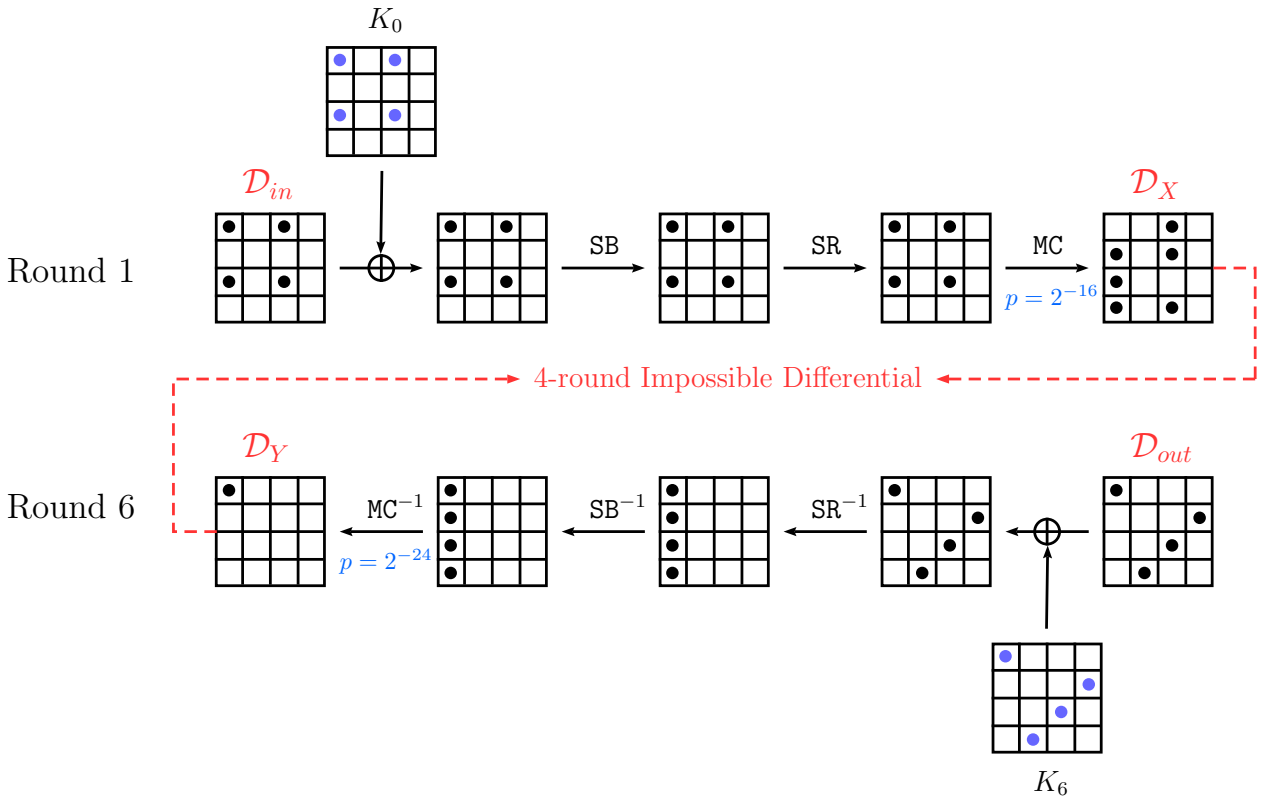
Extend the impossible differential

Example : 6-round attack on AES

- Extend the impossible differential one round to both directions.



Second step: Extend the impossible differential



- Collect pairs verifying $(\mathcal{D}_{in}, \mathcal{D}_{out})$.
- Guess 4 bytes of K_1 and 4 bytes of K_6.
- If for a value of K_1 and K_6 we get at the same time $\mathcal{D}_X$ and $\mathcal{D}_Y$, then these (partial) keys can be discarded.
- Exhaustive search for the remaining candidate keys.

Our contributions

The key recovery phase of impossible differential attacks is a very technical and not fully understood procedure.

- Errors are common.
- Many of the published attacks are sub-optimal.

Our goal :

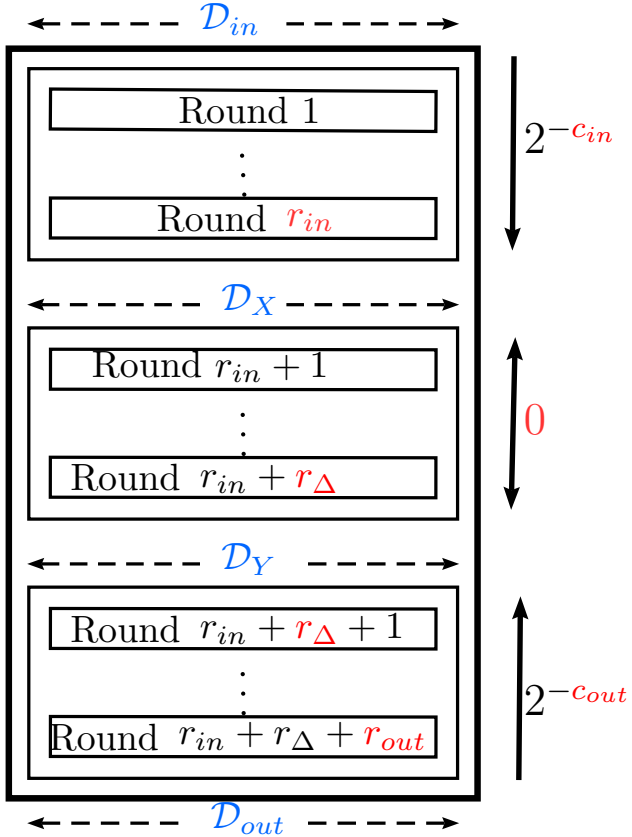
- Formalize the key recovery procedure.
- Provide complete complexity formulas.
- Introduce new techniques for improving the time or data complexity of such attacks.

The results presented next appear in the following two papers:

- *Scrutinizing and Improving Impossible Differential Attacks: Applications to CLEFIA, Camellia, LBlock and Simon*,
  Christina Boura, María Naya-Plasencia and Valentin Suder.
  Presented at ASIACRYPT 2014.

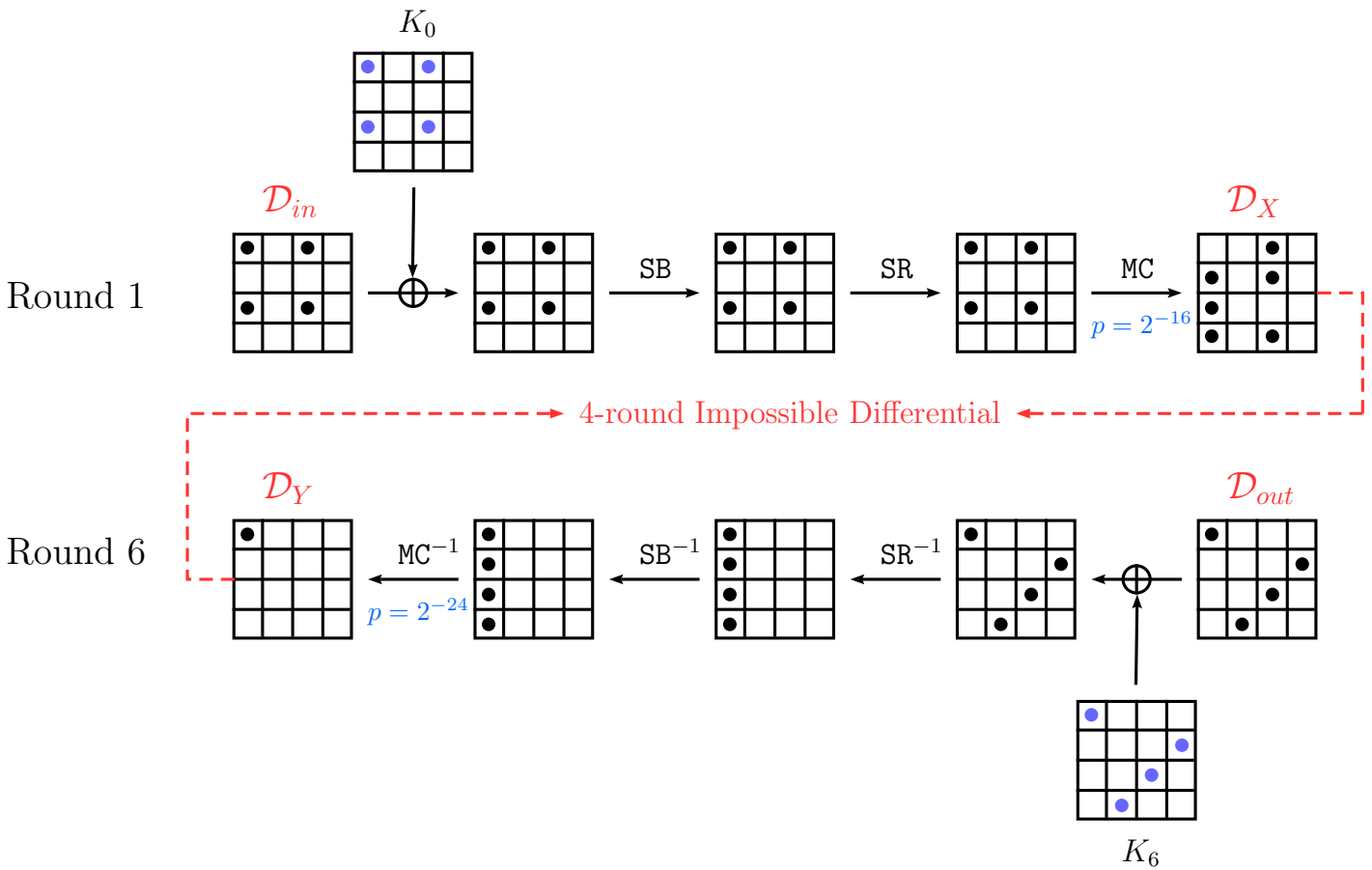
- *Improving Impossible Differential Cryptanalysis*,
  Christina Boura, Virginie Lallemand, María Naya-Plasencia and Valentin Suder, submitted.

Notation



- Δ_V: size in bits of a difference $\mathcal{D}_V$.
- c_{in}, c_{out} : number of bit conditions to be verified.
- k_{in}, k_{out} : number of involved subkey bits.
- $|k_{in} \cup k_{out}|$: key entropy

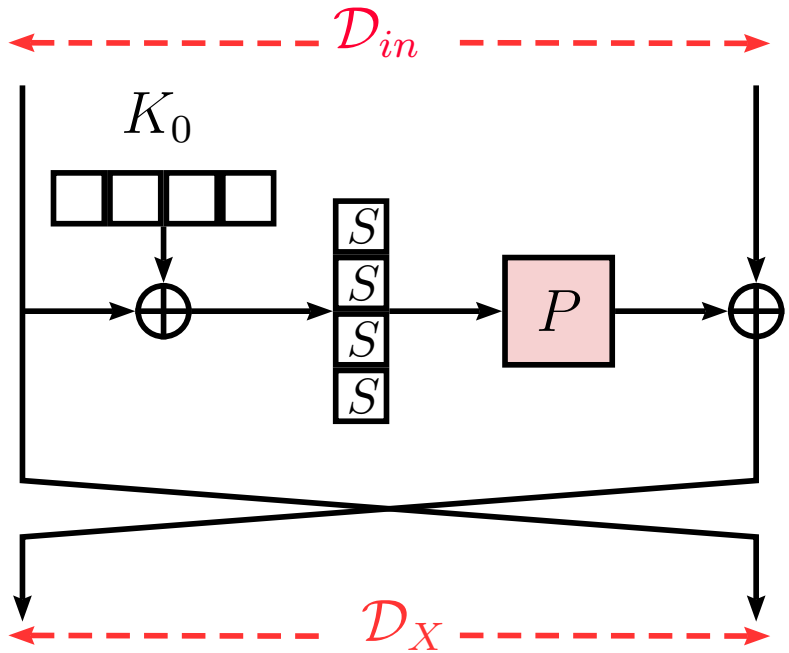
Example



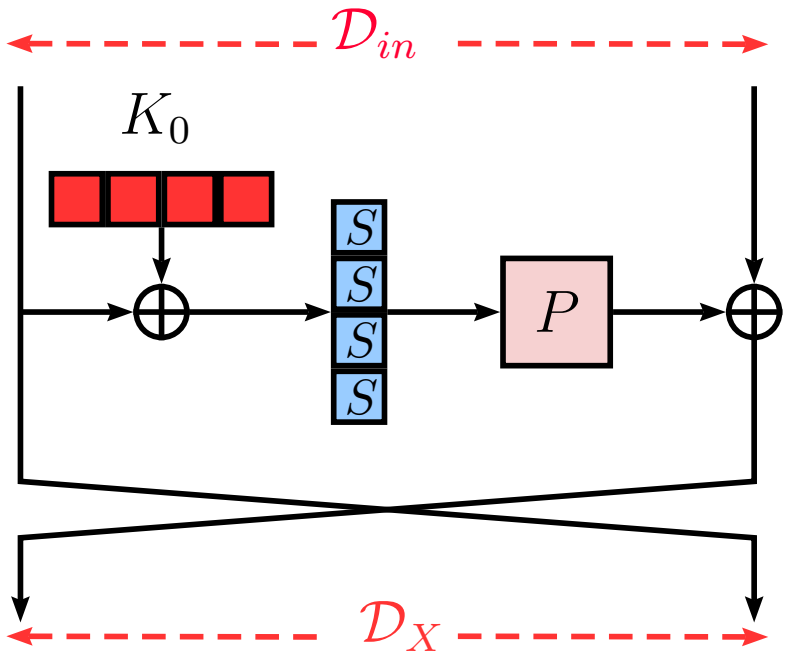
- $\Delta_{in} = 32, \Delta_{out} = 32$
- $c_{in} = 16, c_{out} = 24$
- $k_{in} = 32, k_{out} = 32.$

The early abort technique

Introduced by Lu et al. in 2008 for impossible differential cryptanalysis.

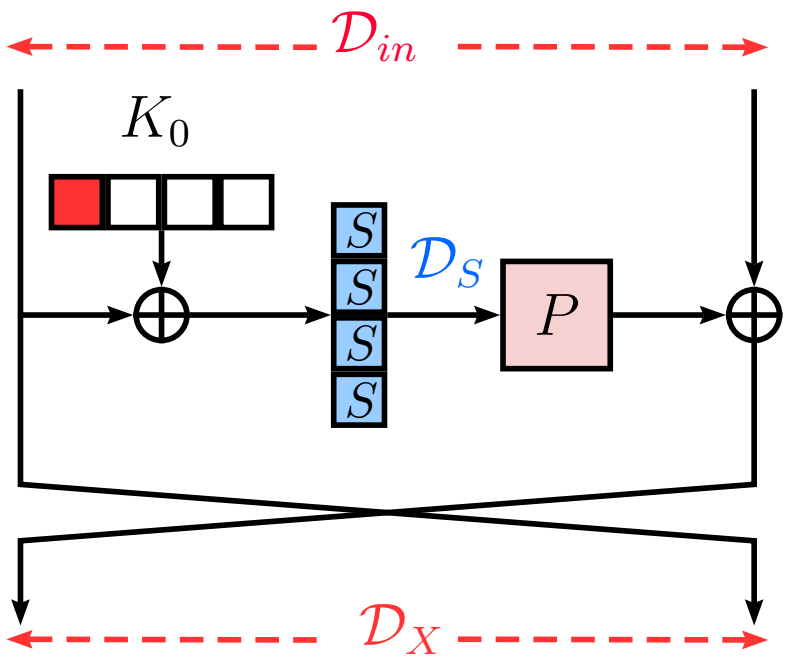


The early abort technique



- **Classical approach**: Guess all the required subkey bits, encrypt a pair and verify if the difference $\mathcal{D}_X$ occurs.

The early abort technique



- **Early abort**: Guess the subkey bits word by word and check if the partial difference occurs.
- If the partial difference doesn't occur, discard the pair.

Outline

How many pairs does an attack require?

By taking N pairs satisfying $(\mathcal{D}_{in}, \mathcal{D}_{out})$, the probability of not discarding a candidate key is

$$P = (1-2^{-(c_{in}+c_{out})})^N$$

How many pairs N are needed for the attack?

- First approach: $(1-2^{-(c_{in}+c_{out})})^N < 2^{-|k_{in} \cup k_{out}|}$
- Better approach: $(1-2^{-(c_{in}+c_{out})})^N < \frac{1}{2}$
- Take

$$N_{\mathsf{min}} = 2^{c_{in}+c_{out}}.$$

Memory complexity : N

Finding N solutions for a given truncated differential

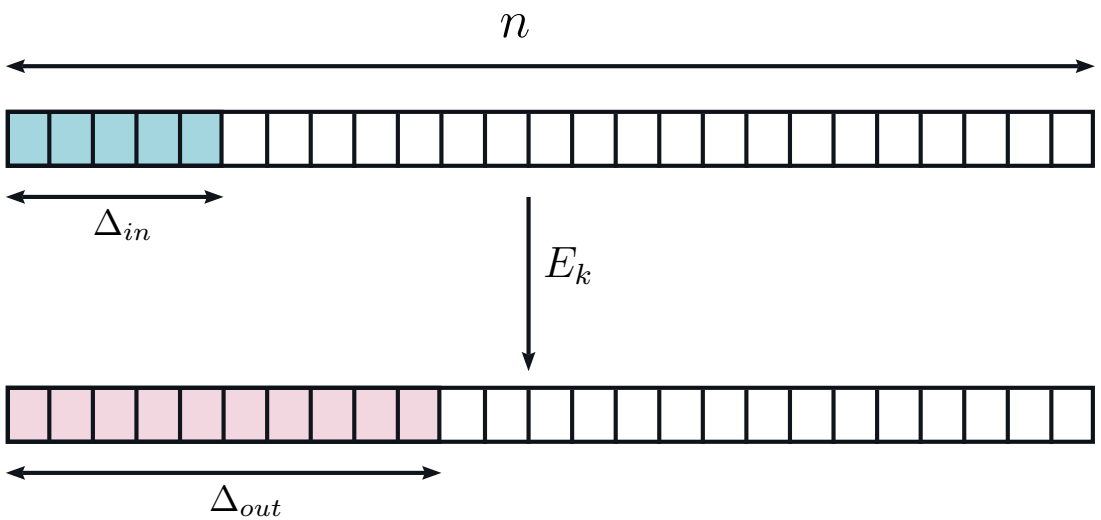
Problem: Find N pairs verifying $\mathcal{D}_{in}$ and $\mathcal{D}_{out}$

- For $N = 1$: Limited birthday technique [Gilbert, Peyrin – FSE 2010]

$$C_1 = \max\{\min_{\Delta \in \{\Delta_{in}, \Delta_{out}\}} \{\sqrt{2^{n-\Delta}}\}, 2^{n-(\Delta_{in}+\Delta_{out})}\},$$

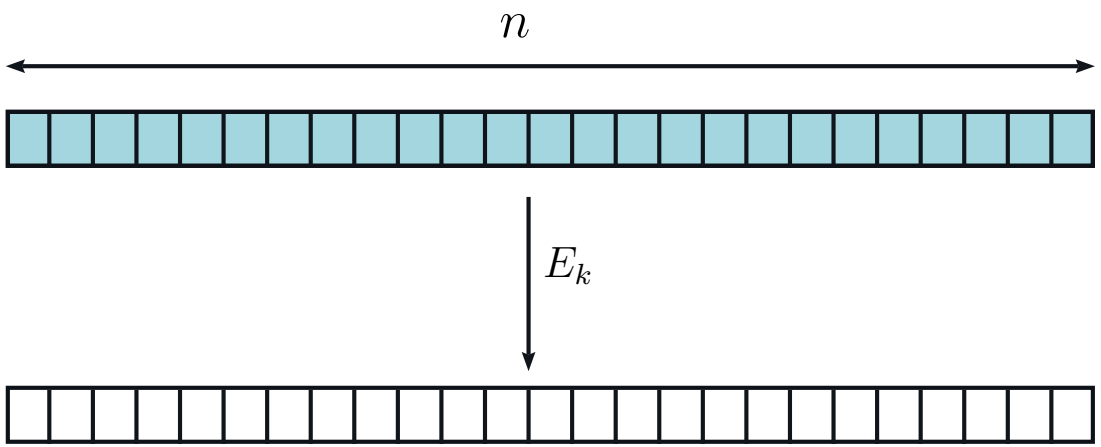
where n is the state size.

Limited Birthday Technique [Gilbert, Peyrin – FSE 2010]



Find a pair of inputs (m, m') such that
$m \oplus m' \in \mathcal{D}_{in}$ and $E_k(m) \oplus E_k(m') \in \mathcal{D}_{out}$

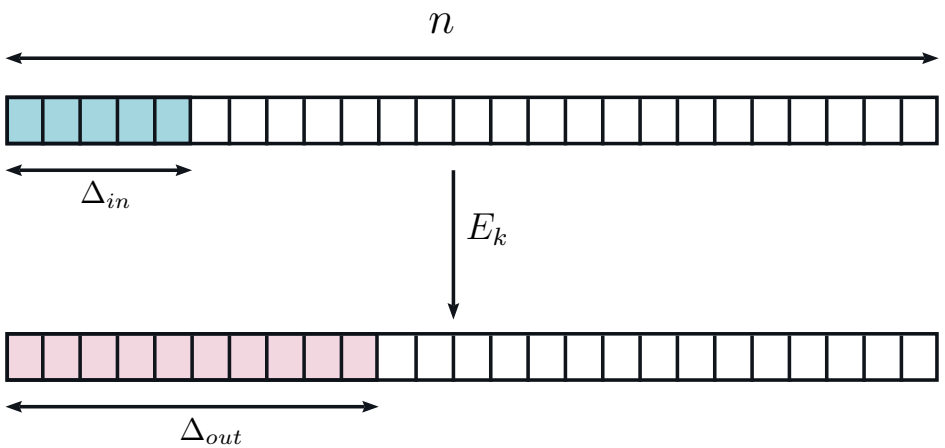
Limited Birthday Technique [Gilbert, Peyrin – FSE 2010]



Extreme case: $\Delta_{out} = 0$ and input unrestricted.

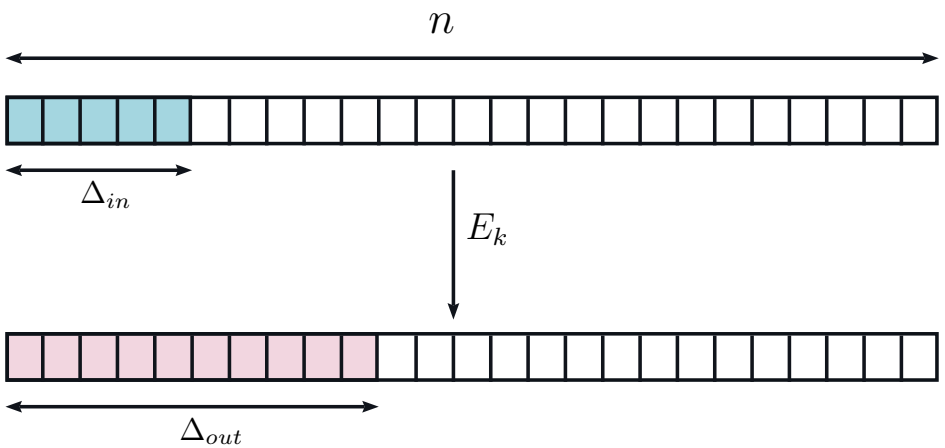
$\Rightarrow$ Collision after $\approx 2^{n/2}$ computations.

Limited Birthday Technique [Gilbert, Peyrin – FSE 2010]



- When the input space is restricted, the number of pairs that can be constructed is reduced.
- Consider the quantity $(2\Delta_{in} + \Delta_{out})$.
- If $(2\Delta_{in} > n - \Delta_{out})$ apply birthday paradox to collide on $n - \Delta_{out}$ bits.
- Else, restart the birthday paradox $2^{n-2\Delta_{in}-\Delta_{out}}$ times.

Limited Birthday Technique [Gilbert, Peyrin – FSE 2010]



$$C_1 = \begin{cases} \sqrt{2^{n-\Delta_{out}}} & \text{if } 2\Delta_{in} > n - \Delta_{out}, \\ 2^{n-(\Delta_{out}+\Delta_{in})} & \text{otherwise.} \end{cases}$$

Cost C_N for finding N pairs verifying $(\mathcal{D}_{in}, \mathcal{D}_{out})$

By considering $C_N = N \times C_1$ we might be wasting some structures.

Determine the number of inputs 2^x we need in order to construct N pairs.

1. $N \leq \frac{2^{\Delta_{in}} 2^{\Delta_{in}-1}}{2^{n-\Delta_{out}}}$ ($\mathcal{D}_{in}$ is large enough). Thus $2^x \leq 2^{\Delta_{in}}$ and therefore $N = \frac{2^{2x-1}}{2^{n-\Delta_{out}}}$. We need

$$2^x = \sqrt{N 2^{n-\Delta_{out}+1}} \text{ inputs.}$$

2. $N > \frac{2^{\Delta_{in}} 2^{\Delta_{in}-1}}{2^{n-\Delta_{out}}}$ ($\mathcal{D}_{in}$ is not large enough). We need to add 2^y structures of size Δ_{in} such that $N = 2^y \frac{2^{\Delta_{in}} 2^{\Delta_{in}-1}}{2^{n-\Delta_{out}}}$. Therefore we need

$$2^x = 2^{y+\Delta_{in}} = N 2^{n-\Delta_{in}-\Delta_{out}+1} \text{ inputs.}$$

Cost for finding N pairs

$$C_N = \max\left\{ \min_{\Delta \in \{\Delta_{in}, \Delta_{out}\}} \left\{ \sqrt{N 2^{n-\Delta+1}} \right\}, N 2^{n-\Delta_{in}-\Delta_{out}+1} \right\}.$$

Data complexity: C_N

Obviously,

$$C_N < 2^n$$

Time complexity

$$T_{\text{comp}} = C_N +$$

- Encrypt data.

Time complexity

$$T_{\text{comp}} = C_N + \left(N + 2^{|k_{in} \cup k_{out}|} \frac{N}{2^{c_{in}+c_{out}}}\right) C'_E$$

- Encrypt data.
- Early abort technique
  - Check each key candidate step by step.
  - Decrease the number of pairs in the list.

Time complexity

$$T_{\text{comp}} = \left(C_N + \left(N + 2^{|k_{in} \cup k_{out}|} \frac{N}{2^{c_{in}+c_{out}}}\right) C'_E + 2^K P\right) C_E$$

- Encrypt data.
- Early abort technique
  - Check each key candidate step by step.
  - Decrease the number of pairs in the list.

  The last term corresponds to $2^K P = 2^{K-|k_{in} \cup k_{out}|} P 2^{|k_{in} \cup k_{out}|}$.
- Test by exhaustive search the remaining keys.

The role of the key schedule

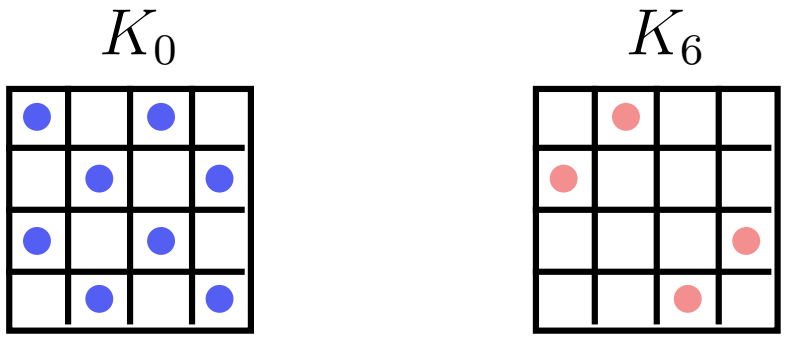
During the key-recovery phase, key bits of different subkeys are guessed.

- How to recover the master key from these guessed bits?

  This depends on the nature of the key-schedule.

- If the key-schedule is (almost) linear, directly translate the k_{in} and k_{out} guessed bits in the same number of bits of the master key.

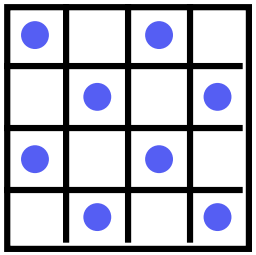
Complex key schedules



If the key-schedule is **complex**, it is not possible to directly translate the information guessed on the subkeys into the same amount of information on the master key.

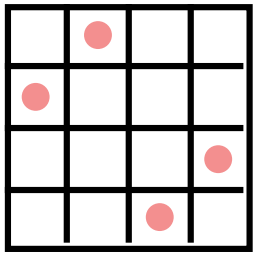
What do we do then?

K_0



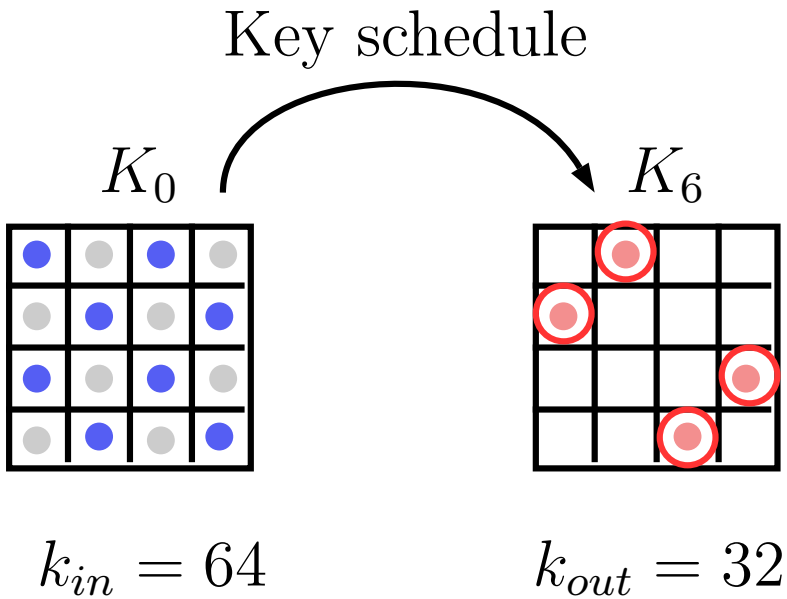
$k_{in} = 64$

K_6



$k_{out} = 32$

What do we do then?



- Complete the missing bits to some of the subkeys.
- Compute through the key schedule.
- Verify if the result matches.

How is the time complexity affected?

A new term has to be added to the time complexity formula.

$$\min(2^{K-k_{in}}, 2^{K-k_{out}}) \cdot P \cdot 2^{k_{in}+k_{out}} C_{KS},$$

where C_{KS} is the key schedule cost.

How is the time complexity affected?

A new term has to be added to the time complexity formula.

$$\min(2^{K+k_{in}}, 2^{K+k_{out}}) \cdot P \cdot C_{KS},$$

where C_{KS} is the key schedule cost.

In previous works, it was wrongly supposed that one guessed word of a subkey could directly be seen as one guessed word of the master key.

Outline

The state-test technique

Goal:

Eliminate some candidate keys without considering all the possibilities for the involved key bits.

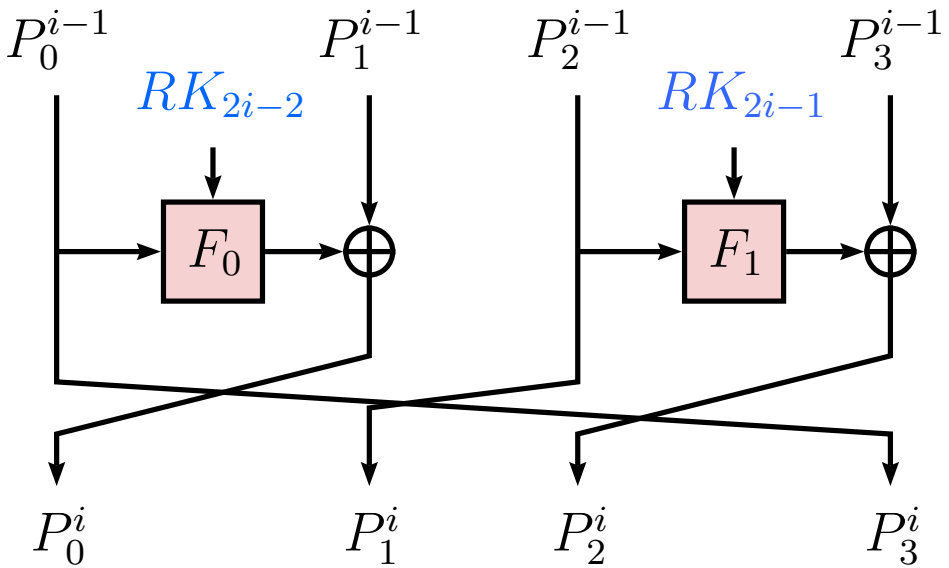
How?

If a word of the state of size s depends on more than s key bits, guess this word instead.

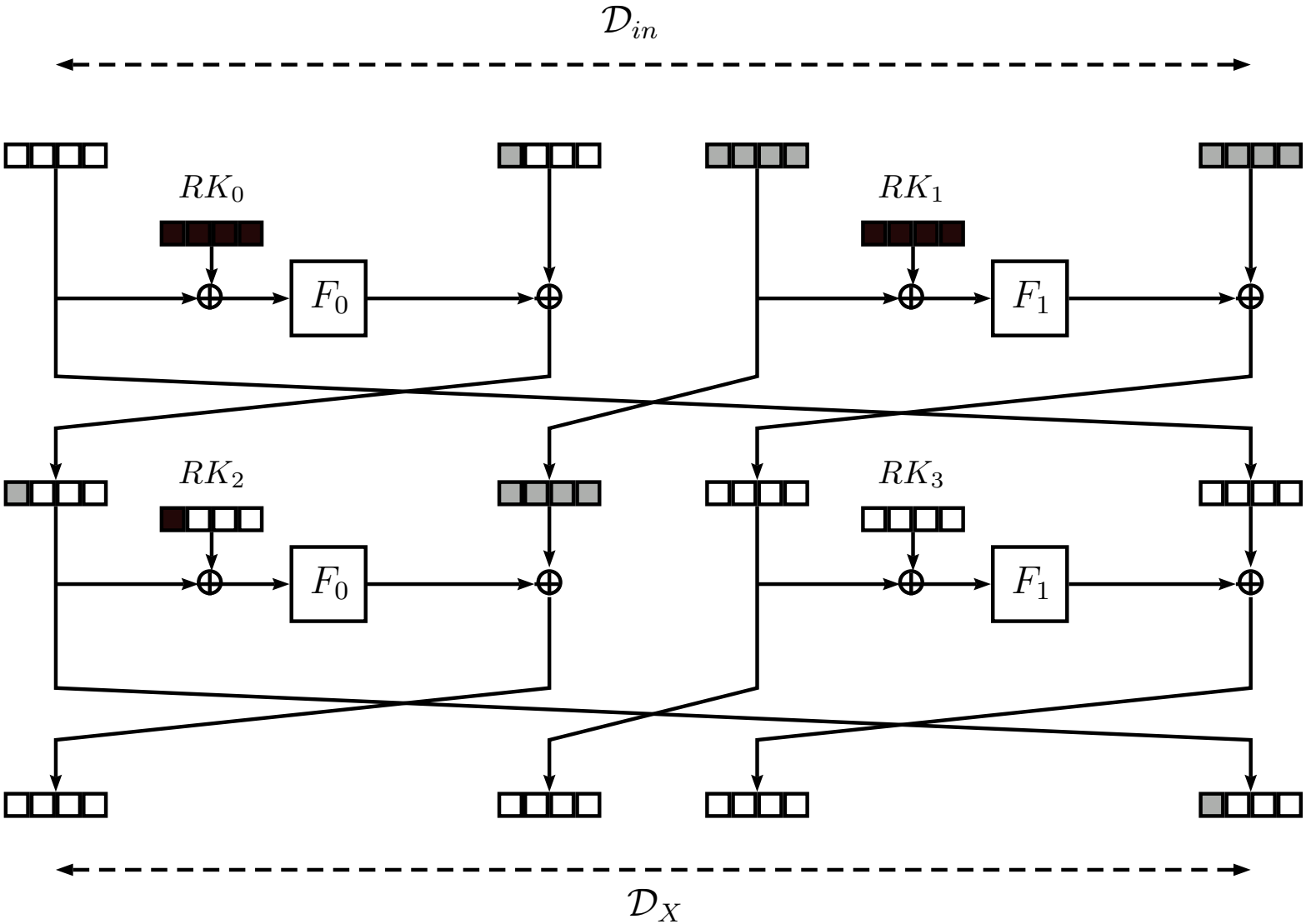
Example on CLEFIA-128

ISO/IEC standard in lightweight crypto. Developed by SONY in 2007.

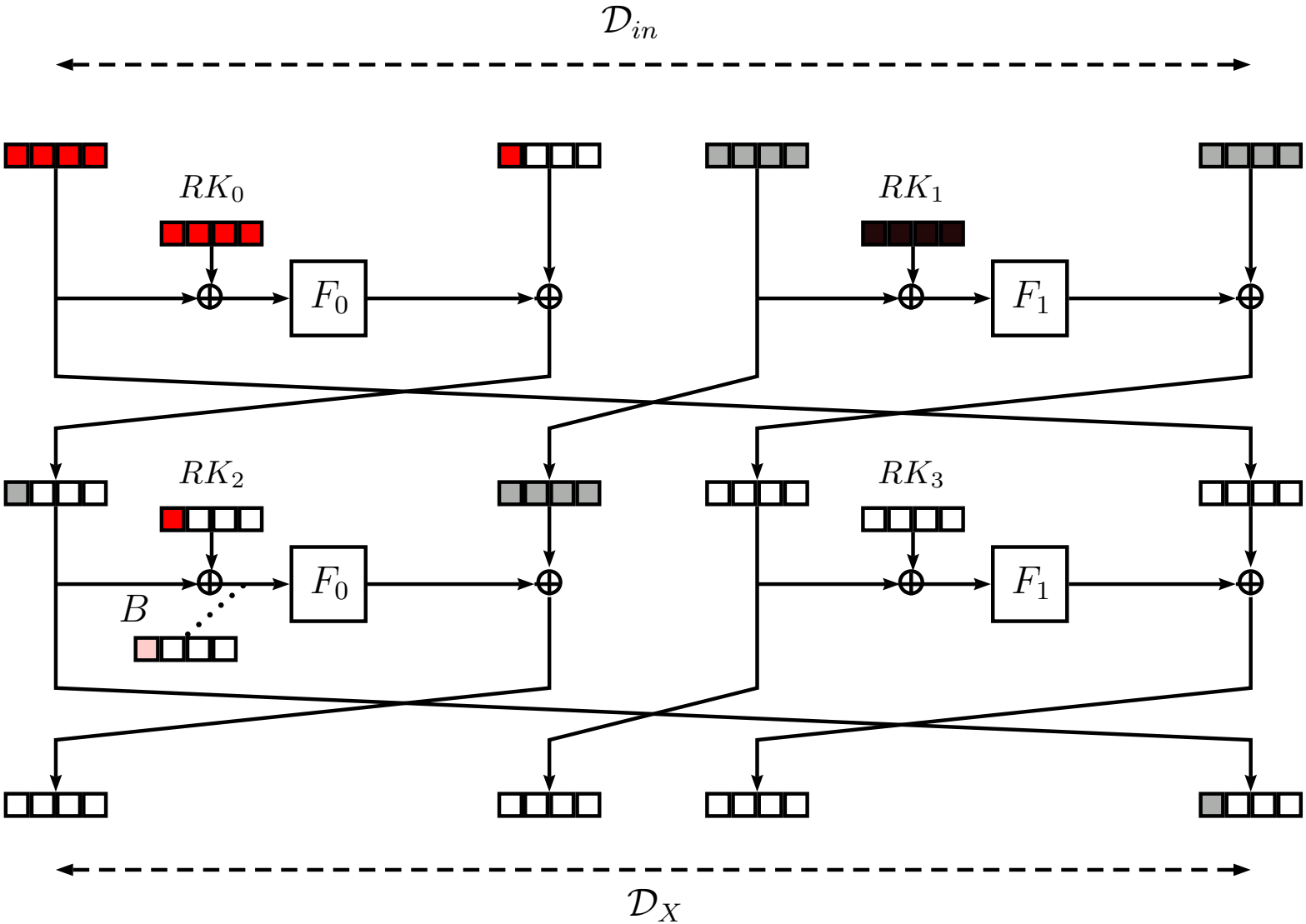
- Block size: $4 \times 32 = 128$ bits
- Key size: 128 bits
- Number of rounds: 18



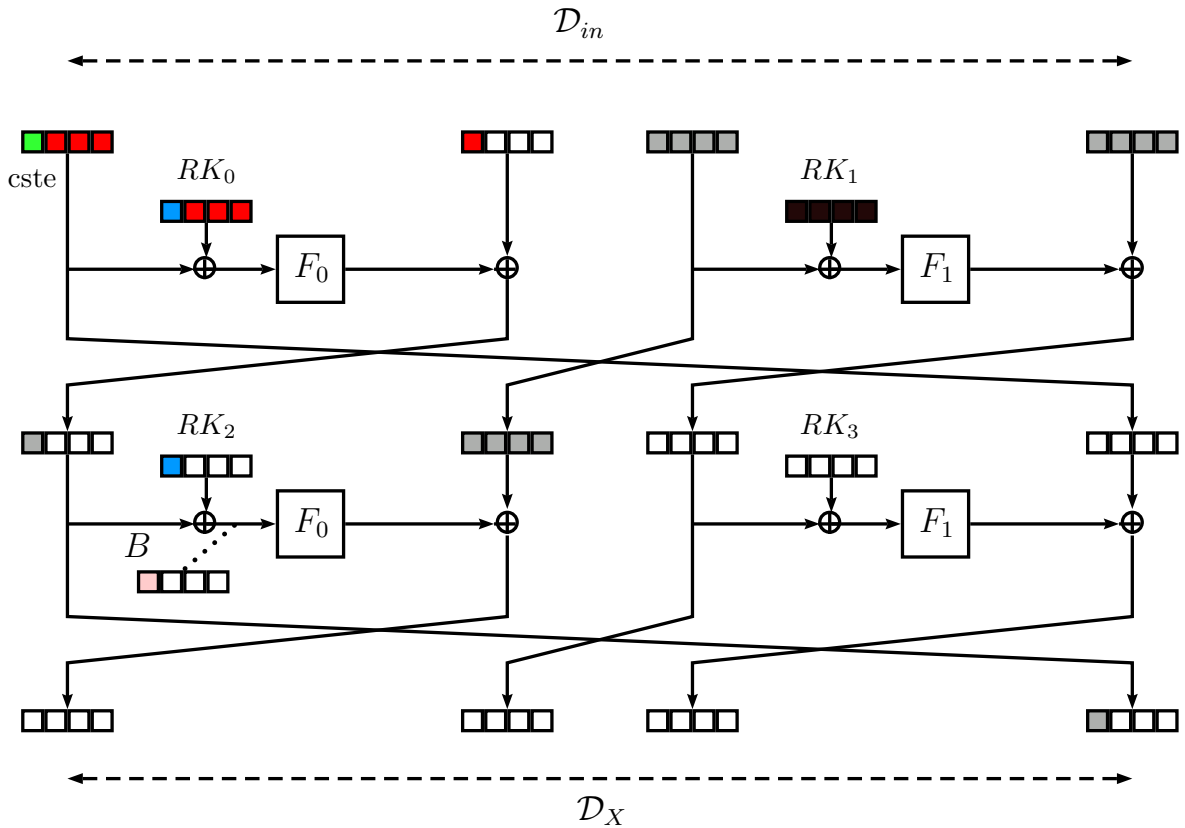
The state-test technique in practice



The state-test technique in practice

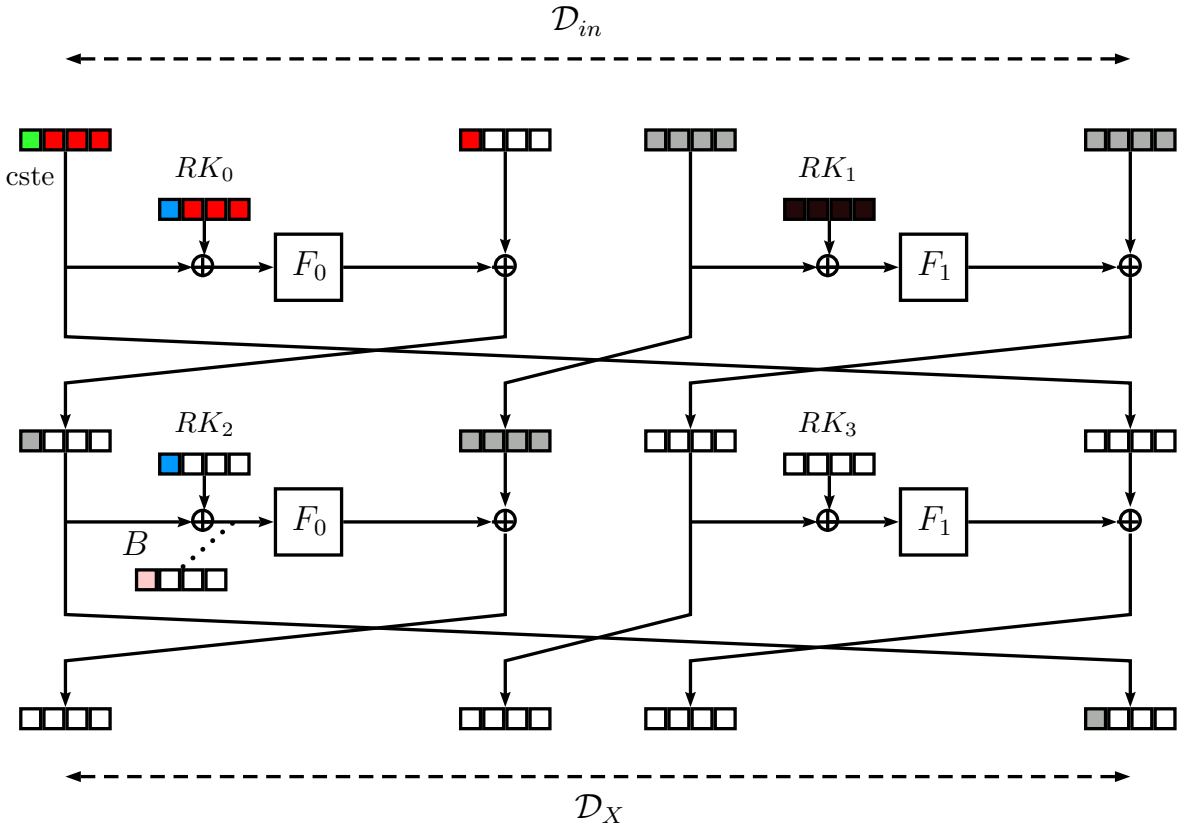


The state-test technique in practice



$$B = \blacksquare \oplus S_0(\blacksquare \oplus \blacksquare) \oplus \blacksquare$$

The state-test technique in practice



$B' = \blacksquare \oplus S_0(\blacksquare \oplus \blacksquare)$ with $B = B' \oplus \blacksquare$

$|k_{in} \cup k_{out}| = 122$ bits $\Rightarrow$ $|k_{in} \cup k_{out}| = 122 - 16 + \underbrace{8}_{B'}$ bits

Remark regarding the state-test technique

- The state-test technique applies to both Feistel and SPN constructions.
- However, it seems to apply better to Feistel ciphers.
  - In SPN ciphers the gain in the time complexity generally leads to an equivalent loss in data complexity, because a part of the active part of the plaintexts has to be fixed.
- We have implemented this technique on a toy-cipher (mini-CLEFIA) and verified its efficiency in practice.

Multiple Impossible Differentials

Formalize the idea of [Tsunoo et al. 08].

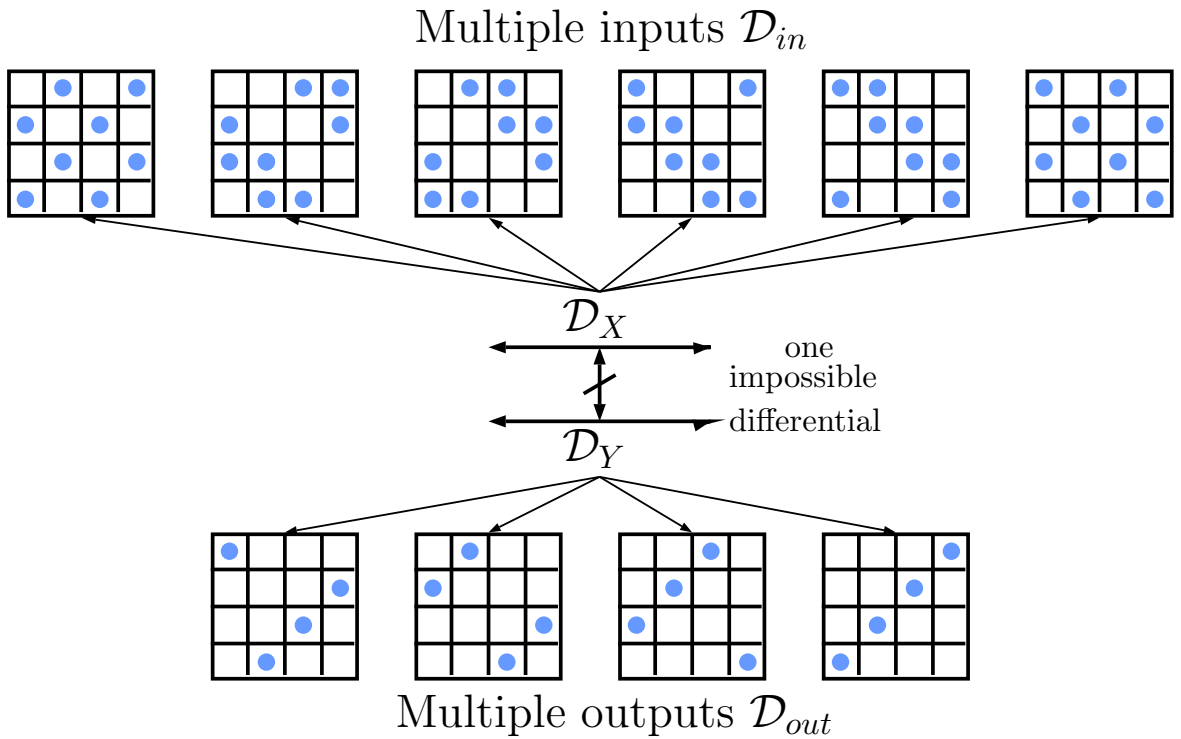
CLEFIA has 9-round impossible differentials $((0,0,0,A) \not\to (0,0,0,B))$ and $((0,A,0,0) \not\to (0,B,0,0))$ when A and B verify:

A	B		
$(0,0,0,\alpha)$	$(0,0,\beta,0)$	$(0,\beta,0,0)$	$(\beta,0,0,0)$
$(0,0,\alpha,0)$	$(0,0,0,\beta)$	$(0,\beta,0,0)$	$(\beta,0,0,0)$
$(0,\alpha,0,0)$	$(0,0,0,\beta)$	$(0,0,\beta,0)$	$(\beta,0,0,0)$
$(\alpha,0,0,0)$	$(0,0,0,\beta)$	$(0,0,\beta,0)$	$(0,\beta,0,0)$

$$C_N = 2^{113} \quad \Rightarrow \quad C_N = 2^{113-\log_2(24)}$$

Multiple differentials in impossible differential cryptanalysis

- More choices for the input/output patterns of a pair.
- Less data is needed to construct the pairs for the attack $\rightarrow$ reduction in the data complexity

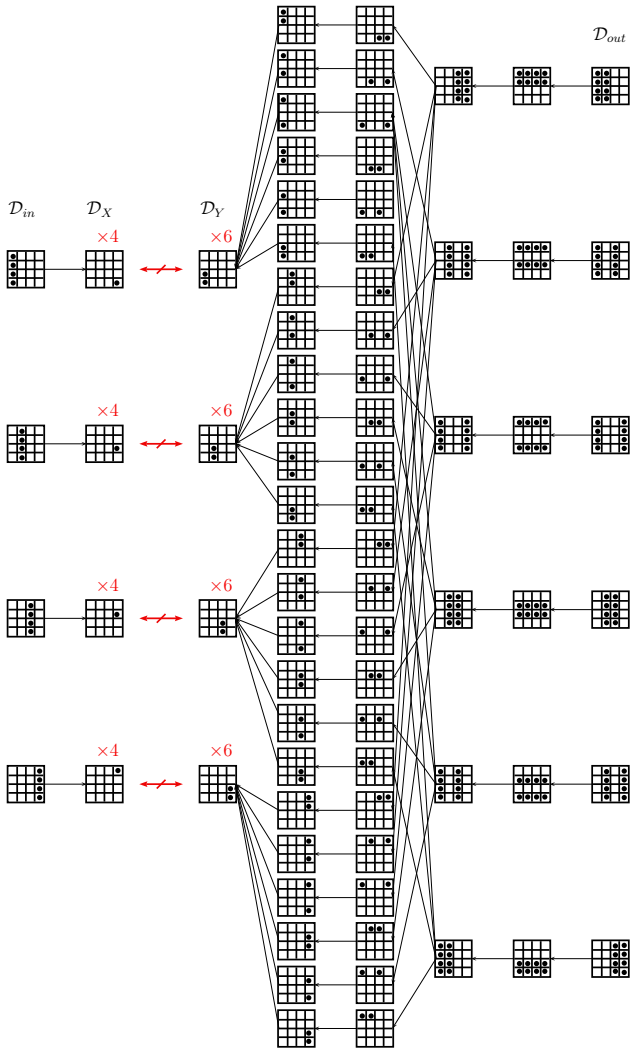


- The $\log_2$ of the data complexity is reduced by

  # input multiples + # output multiples.

Combine multiple impossible diff. with multiple diff.

Use multiple differentials and multiple impossible differentials together to further reduce the amount of data.



Outline

Applications

Feistel ciphers

- Best cryptanalysis on CLEFIA-128, Camellia and LBlock.
- Best impossible differential attacks on the SIMON family.

SPN ciphers

- Best impossible differential attacks on AES-128, CRYPTON-128 and ARIA-128.
- Each application illustrates a different combination of the new techniques.

Results on Feistel ciphers

Algorithm	Rounds	Data (CP)	Time	Memory (Blocks)	Tech.	Ref.
CLEFIA-128	13	$2^{117.8}$	$2^{121.2}$	$2^{86.8}$	ID	[MDS 11]
	13	$2^{114.4}$	$2^{114.4}$	2^{80}	ID	
Camellia-128	11	2^{122}	2^{122}	2^{98}	ID	[LLGWLCL 12]
	11	$2^{118.4}$	$2^{118.43}$	$2^{92.4}$	ID	
Camellia-192	12	2^{123}	$2^{187.2}$	$2^{155.41}$	ID	[LLGWLCL 12]
	12	$2^{119.7}$	$2^{161.06}$	$2^{150.7}$	ID	
Camellia-256	13	2^{123}	$2^{251.1}$	2^{203}	ID	[LLGWLCL 12]
	13	$2^{119.71}$	$2^{225.06}$	$2^{198.71}$	ID	
Camellia-256‡	14	2^{120}	$2^{250.5}$	2^{120}	ID	[LLGWLCL 12]
	14	$2^{117.7}$	$2^{215.7}$	$2^{166.7}$	ID	
LBlock	22	2^{58}	$2^{79.28}$	$2^{72.67}$	ID	[KDH 12]
	23	$2^{63.87}$	$2^{74.30}$	2^{60}	ZC	[BM 14]
	23	$2^{55.5}$	2^{72}	2^{65}	ID	

Results on SPN ciphers

Algorithm	Rounds	Data (CP)	Time	Memory (Blocks)	Tech.	Ref.
AES-128	7	$2^{106.2}$	$2^{110.2}$	$2^{90.2}$	ID	[MDRM 10]
	7	2^{105}	$2^{105} + 2^{99}$	2^{90}	MITM	[DFJ 13]
	7	2^{97}	2^{99}	2^{98}	MITM	[DFJ 13]
	7	$2^{113.1}$	$2^{113.1} + 2^{105.1}$	$2^{74.1}$	ID	
	7	2^{105}	$2^{106.88}$	2^{74}	ID	
CRYPTON-128	7	2^{97}	$2^{97.2}$	2^{100}	Tr. Diff.	[KHLSY 03]
	7	2^{121}	$2^{121} + 2^{116.2}$	2^{119}	ID	[MSD 10]
	7	$2^{114.92}$	$2^{114.92} + 2^{113.7}$	$2^{88.5}$	ID	
	8	2^{126}	$2^{126.2}$	2^{100}	Tr. Diff.	[KHLSY 03]
ARIA-128	6	2^{113}	$2^{121.6}$	2^{113}	ID	[LSZL 08]
	6	2^{121}	$2^{121} + 2^{112}$	2^{121}	ID	[WZD 07]
	6	2^{120}	$2^{120} + 2^{96}$	2^{120}	ID	[LS 08]
	6	2^{111}	$2^{111} + 2^{82}$	2^{71}	ID	
	7	$2^{105.8}$	$2^{105.8} + 2^{100.99}$	$2^{79.73}$	LC	[LGLLL 11]

Conclusions

- The proposed techniques are general, however the level of applicability of each method is different on SPN and Feistel ciphers.
- Important to verify the new techniques by implementing them.
- Apply the same approach to other families of attacks, e.g. zero-correlation attacks.

Conclusions

- The proposed techniques are general, however the level of applicability of each method is different on SPN and Feistel ciphers.
- Important to verify the new techniques by implementing them.
- Apply the same approach to other families of attacks, e.g. zero-correlation attacks.

Thanks for your attention!