Meet-in-the-Middle Through an Sbox

A. Canteaut, M. Naya-Plasencia, B. Vayssière
SECRET, INRIA; UVSQ, France

(On going work)

Outline

- Improvement of mitm attacks (**more rounds**)

- Applications so far:
  - Illustrative ex: Present (7 and 9 rounds)
  - DES: 7 rounds (1 more)
  - PRINCE: 8 r, 9r?, 10r??
  - AES bicliques: improved constant (5 less sboxes)

- Questions raised and some answers.

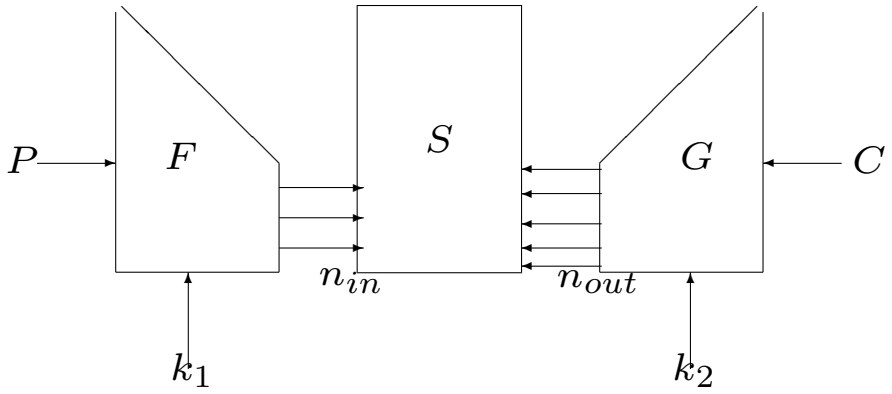
Mitm through an Sbox

Meet-in-the-Middle Attacks

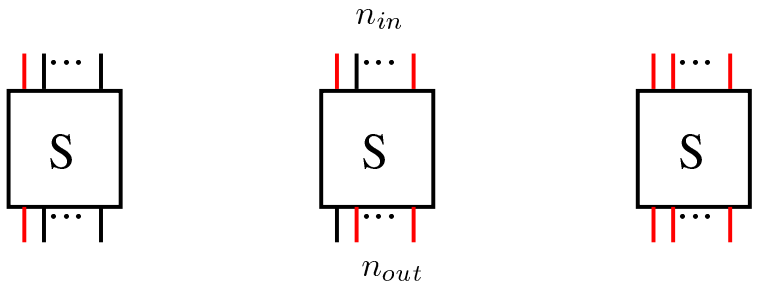
- Largely applied tool.
- Many improvements: partial matching, multidimensional, bicliques...
- Related to some results on hash functions (Aoki and Sasaki - preimage Md5, Khovratovich et al. - Luffa analysis.)

Meet-in-the-Middle Through an Sbox

- Instead of computing the exact same part of the state forward and backward, we compute some inputs to an Sbox S, and some outputs, so that they allow us to perform sieving.
- What can S be? When can we sieve?



Example: $m \times m$ Sbox. How do we sieve?



- Transition exists with probability p.
- n_{in} fixed bits, at most $2^{m-n_{in}}$ values for the n_{out} bits.

$$p \leq \frac{2^{m-n_{in}}}{2^{n_{out}}}$$

- A priori, to sieve we need $p < 1 \Rightarrow n_{in} + n_{out} > m$
- Look for transitions instead of collisions.

What is S?

- It can basically be anything.
- We just need to be able to precompute and store the possible transitions (in the case of a classical Sbox, just the Sbox itself).
- Next we get a list of inputs forward and a list of outputs backward: and merge both with the middle conditions (for ex.: N-P 2011).

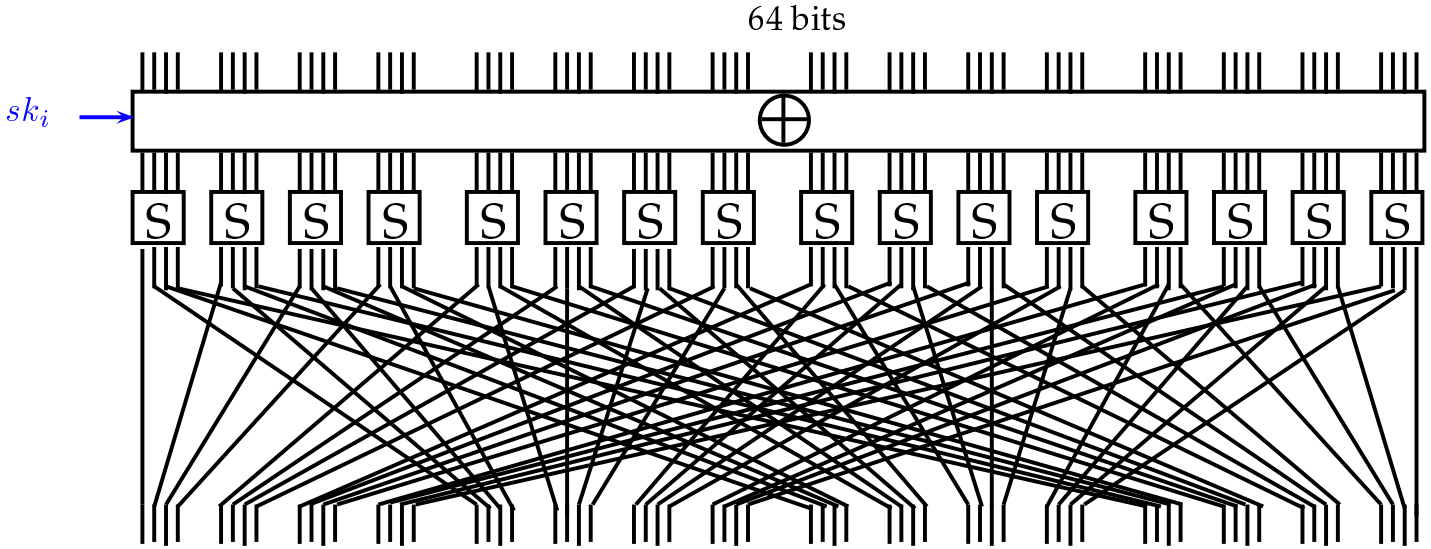
Applications so far

- PRESENT: Representative. Implemented.
- DES: $+1$ round from 1st one (7 rounds). Implemented
- PRINCE: 8 rounds, 9? rounds, 10? looks difficult.
- AES-biclique: tiny constant improvement (log from $126.14 \Rightarrow 125.97$) with a precomputation/storage of 2^{32} (just saying to keep this new improvement in mind.)

Illustrative example: Application to PRESENT-80

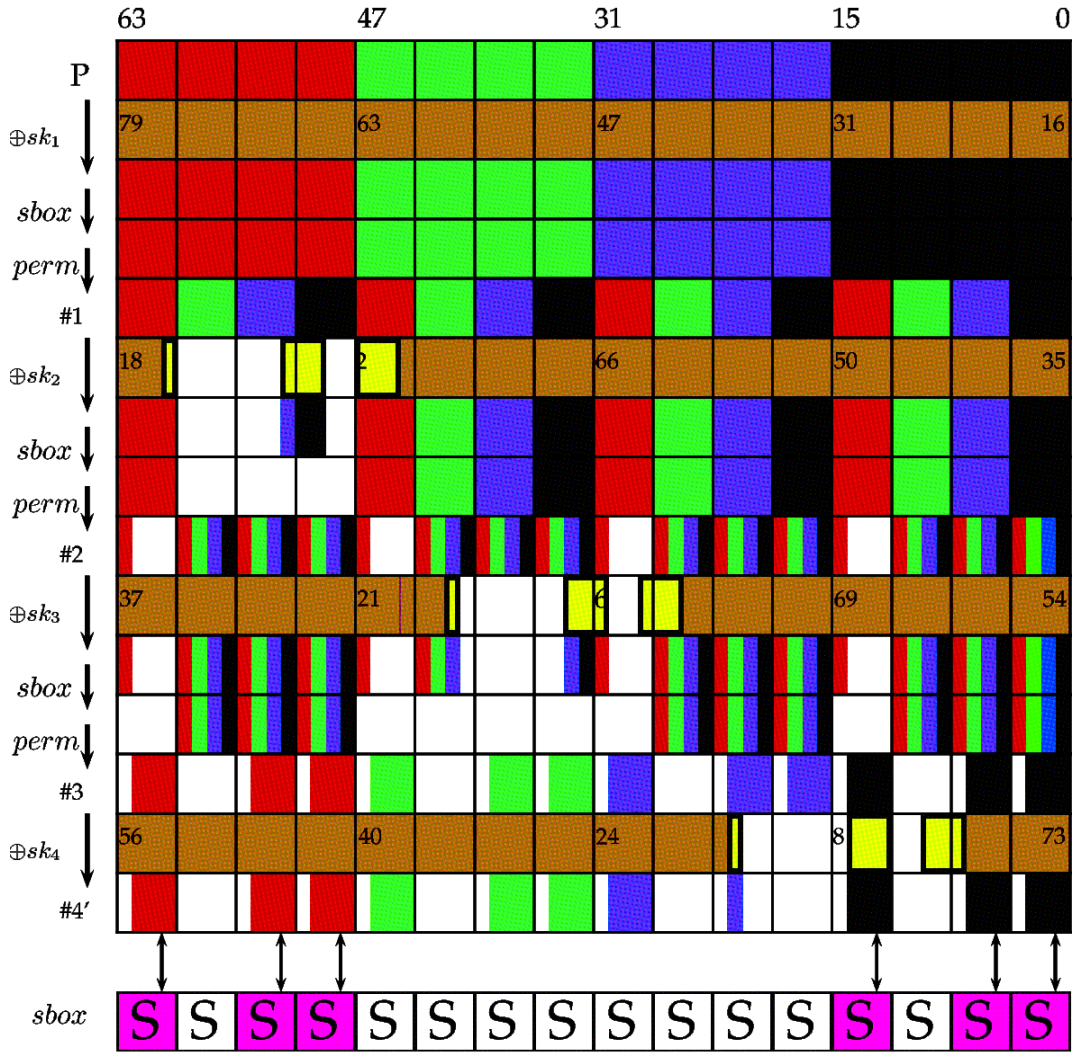
PRESENT[Bogdanov et al. 2007]

Block $n = 64$ bits, key 80 or 128 bits.

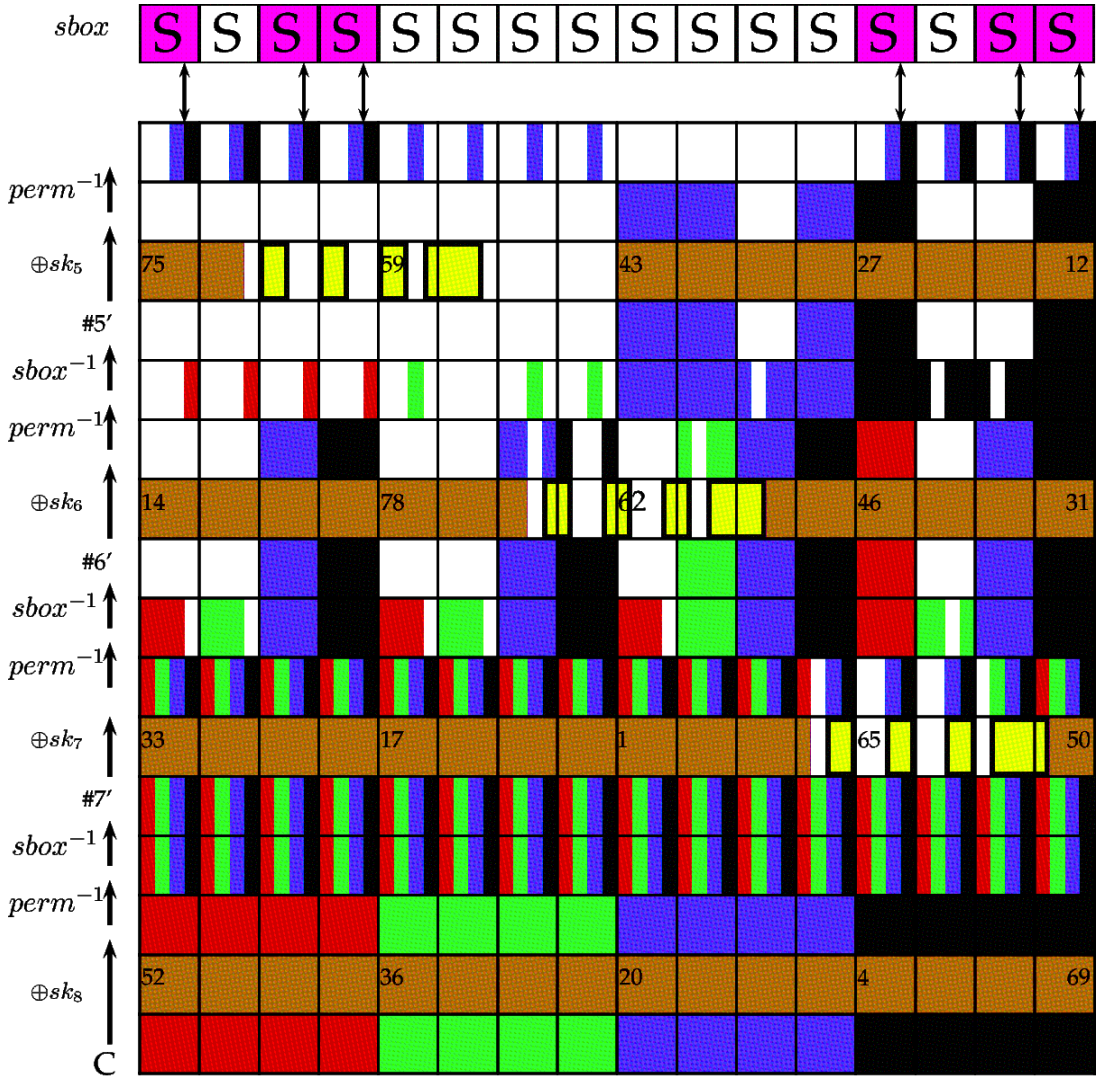


31 rounds + 1 key addition.

Forward Computation



Backward Computation



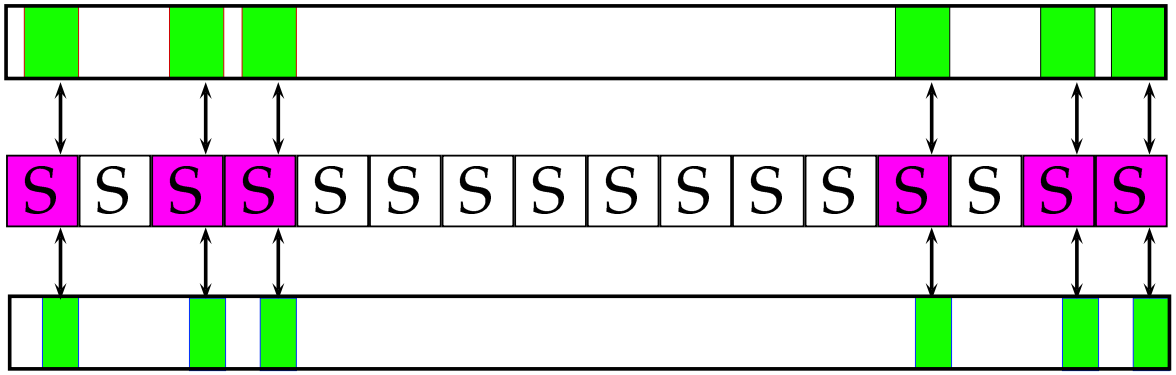
Sieving through the Sboxes: 1 Sbox

$x_3x_2x_1x_0$	$S(x)_3S(x)_2S(x)_1S(x)_0$
0000	1100
0001	0101
0010	0110
0011	1011
0100	1001
0101	0000
0110	1010
0111	1101
1000	0011
1001	1110
1010	1111
1011	1000
1100	0100
1101	0111
1110	0001
1111	0010

$x_2x_1x_0 \rightarrow_S y_1y_0$
000→00
000→11
001→01
001→10
010→10
010→11
011→00
011→11
100→00
100→01
101→00
101→11
110→01
110→10
111→01
111→10

16 values of x_2, x_1, x_0, y_1, y_0, out of 32, correspond to a valid transition.

Sieving through the Sboxes



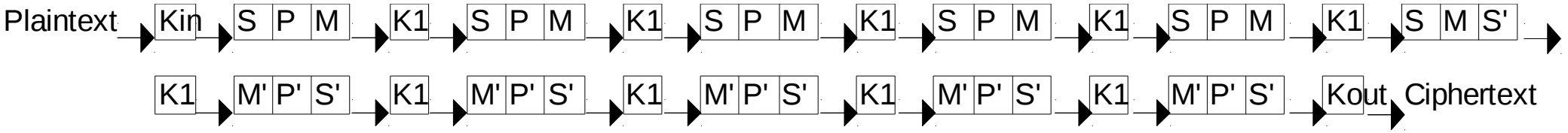
- Probability for 1 Sbox $p = 16/32 = 1/2$
- Probability for the 6 Sboxes: $\frac{1}{2^6}$
- We only try $2^{80-6} = 2^{74}$ potential key candidates.
- 7 rounds.

Application to PRINCE

PRINCE[Borghoff et al. 2012]

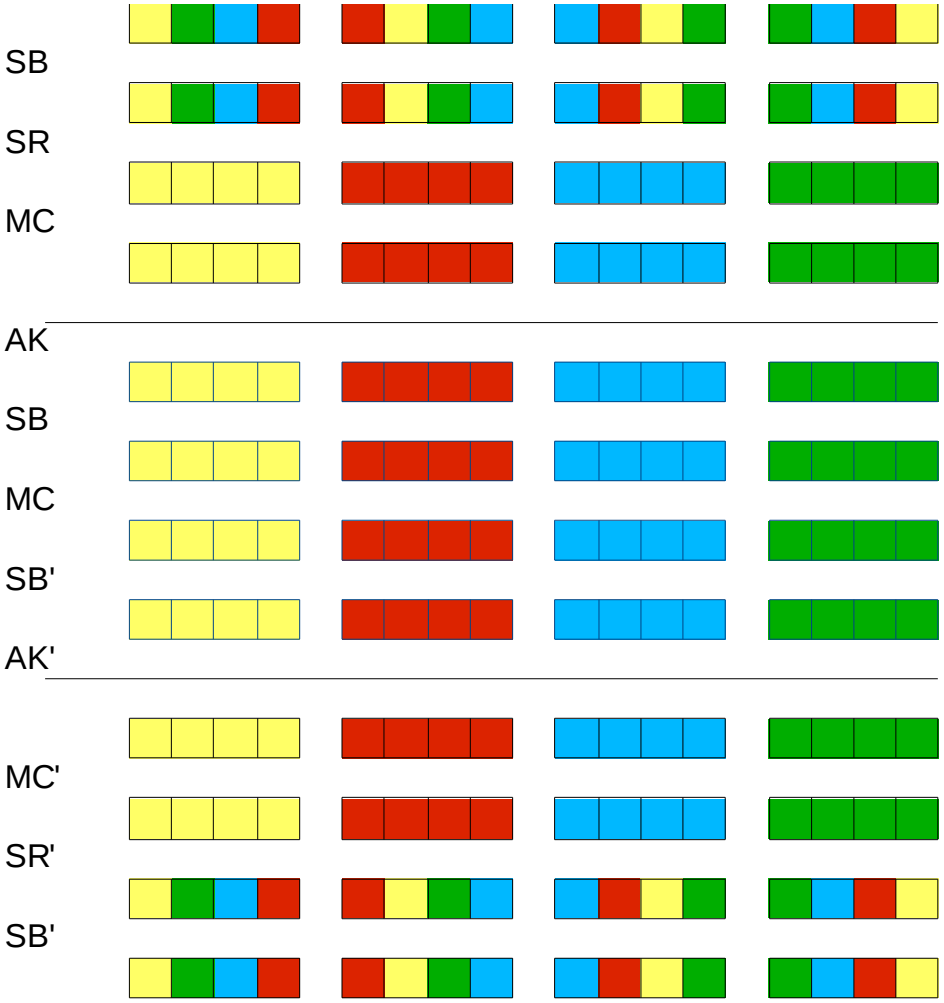
Block cipher 64 bits. $|K1| = |K0| = 64$ (128 keybits).
Best attack so far on 6 rounds out of 12?.

- Non-linear layer of 16 4x4 Sboxes (S).
- Linear layers: permutation of nibbles (P) and "mixcolumns" on groups of 4 nibbles(M).



Kin=K1+K0; Kout=K1+(K0<<<1+K0>>>63).

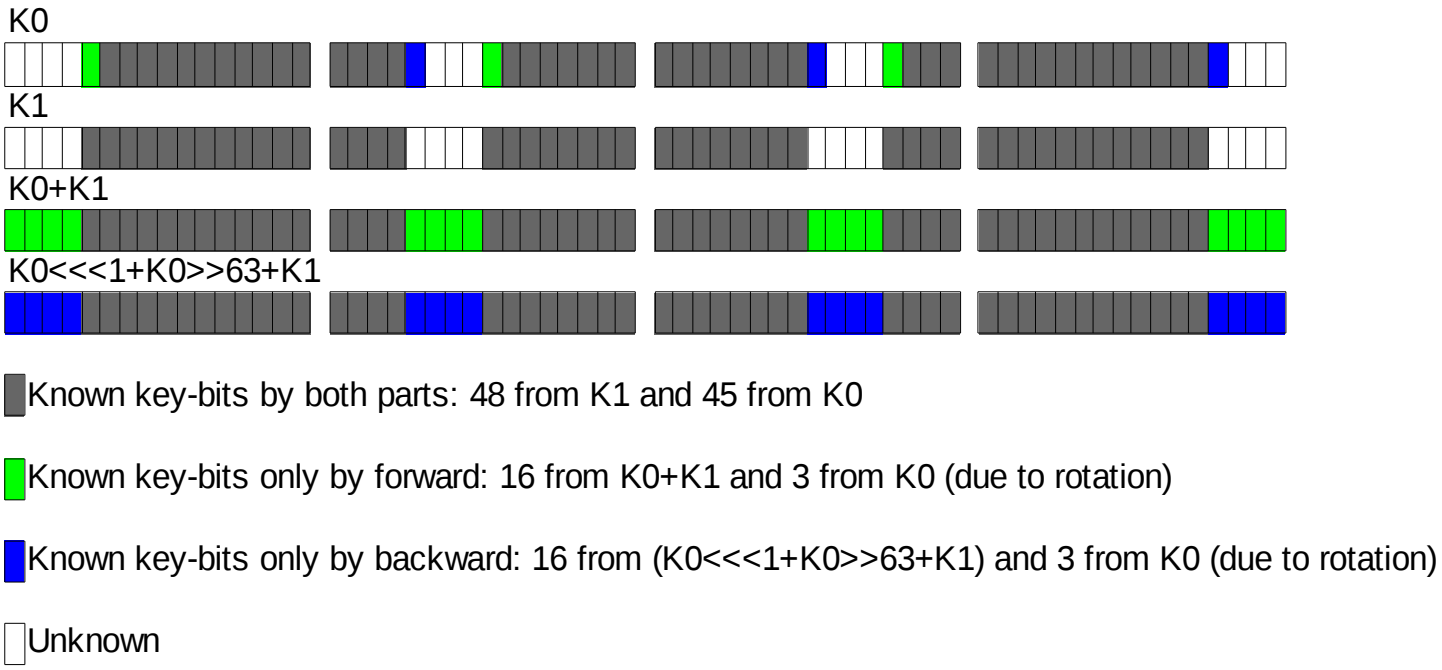
Defining an Sbox over 4 rounds



S

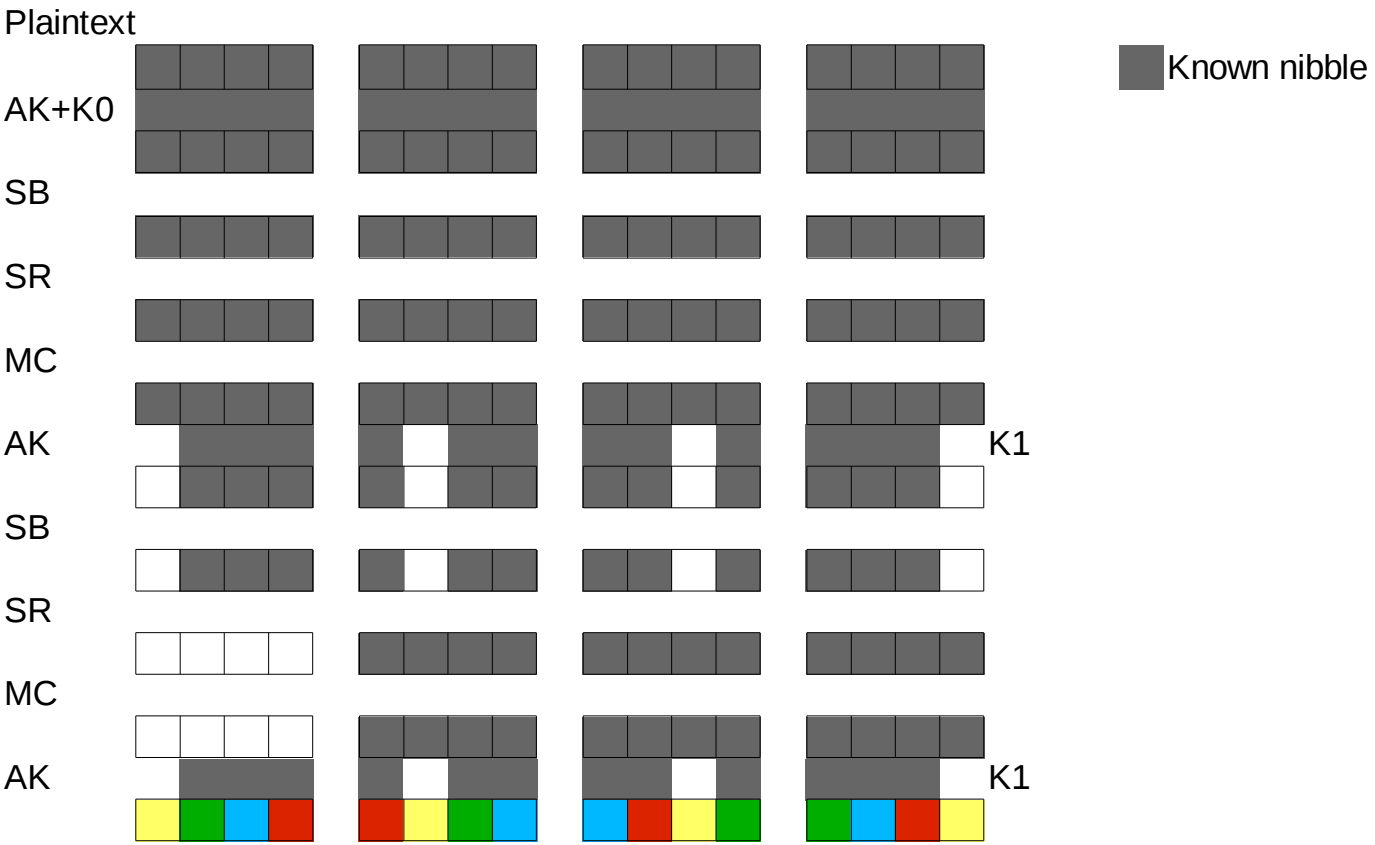
(4 Sboxes 16x16: yellow,red,blue and green), each involving 16 different bits of K1.

Guessing keybits

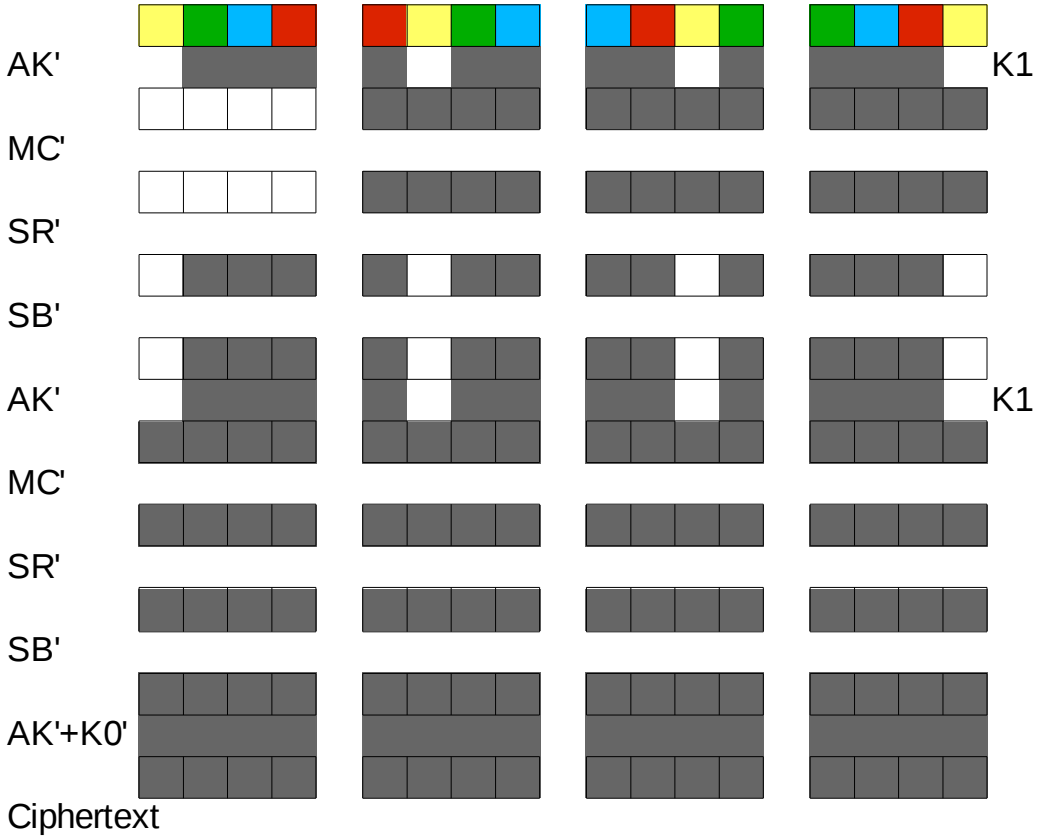


Complexity: $2^{48}2^{45}(2^{3}2^{16} + 2^{3}2^{16})$

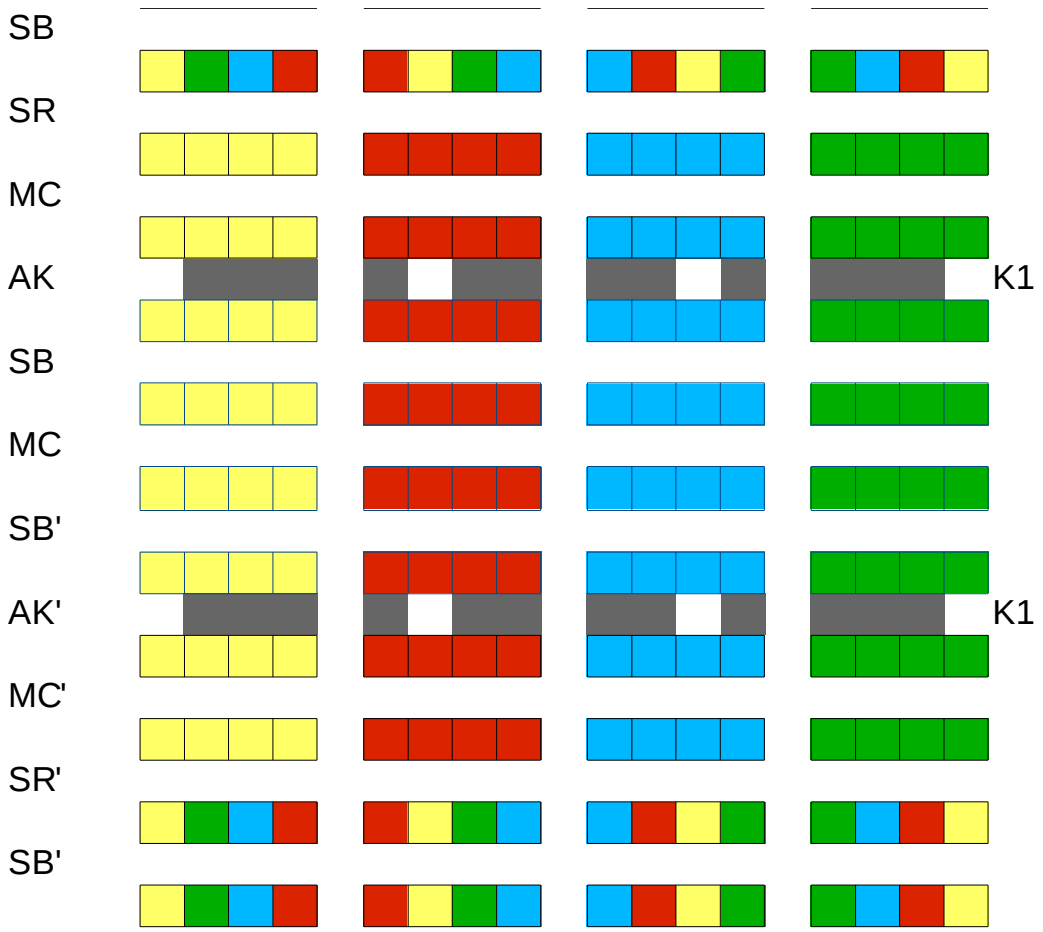
Forward computation



Backward computation



MITM through an Sbox (Precomputed)



Complexity , more rounds?

- We known 12 input bits and 12 output bits, with 4 keybits missing per Sbox: sieve of $16+4-12-12=4$ bits per 16 bit sbox (3 involved).

- $2^{48}(2^{20}+2^{45}(2^{19}+2^{19}+2^{19+19-12}))\approx 2^{119}$ in time, 2^{20} in memory and 1 pair of plaintext-ciphertext for recovering the 128-bit key in 8 rounds.

- Improvement of the time complexity: more sieving derived from K0 conditions.

Questions and some answers

Questions arising

- Which is the min value of n_{in} and n_{out} for sieving?(not always need for $n_{in} + n_{out} > m$)
- What happens for non-bijective Sboxes?
- Can we say 'a priori' how much can we sieve?
- Which Sboxes provide the best resistance?

Some answers (1)

- Lower bound on the minimal value of $(n_{in} + n_{out})$ for sieving from the dual distance of the nonlinear code $(x, S(x))$. In particular, there exist some sieving input and output sets with $n_{in} + n_{out} < m$ if and only if the code is not MDS.

- The sieving probability $p = 2^{m-(n_{in}+n_{out})}$ for all input and output sets if $n_{in} + n_{out} > 2m - d_{min}$ where d_{min} is the minimum distance of the code $(x, S(x))$ (i.e., the branch number of the Sbox).

Some answers (2)

- When $n_{out} = 1$, if there exists a sieving for n_{in} then

$$\mathcal{L}(S) \geq 2^{m-n_{in}}.$$

- When $n_{in} = m - 1$ or when $n_{in} + n_{out} \geq m$ we can compute the efficiency of a particular sieve from the differential table. (If $< m$ some particular cases).

Combinations for Improvements

- Several pairs $\Rightarrow$ more sieving, lower complexity.
- "Clever" lists in the middle for improving the sieve?
- Possible to combine it with bicliques: for ex, $+1$ round for PRESENT without extra complexity.
- Possible improvements using guess of intermediate state bits like Dunkelman et al. 2007

Conclusion[1]

Generic improvement of MITM attacks $\Rightarrow$ potentially more rounds.

- Applications: PRESENT(8 and 10 rounds),DES(7 and 8? rounds), PRINCE(8, 9? rounds), AES-biclique(tiny constant improvement).
- Theoretical questions raised (+ some answers).

Some notions about how to design the sbox, and how to know what to look for when performing the forward and backward computations.

[1]Special thank you to the snow fighters.